GRAND ALLIANCE HDTV SYSTEM SPECIFICATION. PASSAGE TEXT

NPL reference number: XP002055175

Publication date: 1994-02-22

Author: NATIONAL ASSOCIATION OF BROADCASTERS (NAB)

Editor: NATIONAL ASSOCIATION OF BROADCASTERS (NAB)

Classification:

- European: H04N7/26S1

Publication data: GRAND ALLIANCE HDTV SYSTEM SPECIFICATION. DRAFT DOCUMENT SUBMITTED TO THE ACATS TECHNICAL SUBGROUP, FEB. 22, 1994. REPRINT FROM PROCEEDINGS OF THE ANNUAL BROADCAST ENGINEERING CONFERENCE, LAS VEGAS, MAR. 20 -24, 1994, PROCEEDINGS OF THE ANNUAL BROADCAST ENGINEERING CONFERENCE, - ---, NAB, US

Source information: Vol: CONF. 48, Page(s): A,I-III,1-64

Publisher accession number:

Patent applicant:

Publication number:

Report a data error here

Abstract not available for XP002055175

Data supplied from the ***esp@cenet*** database - Worldwide

P.D. 22-02-1994
p. A+I-III +1-64 = 68.

Table of Contents

SCOPE OF DOCUMENT

This document details the System Specification of the Grand Alliance HDTV System, and is intended to form the basis for the documentation of the proposed standard. It is comprised of the documents that have evolved from the work of the Grand Alliance Specialist Groups with the guidance and cooperation of the Advisory Committee on Advanced Television Services (ACATS) Experts Groups. This system specification document also details the prototype hardware, currently under construction. This prototype implementation will be delivered to the Advanced Television Test Center (ATTC) for verification of the performance of the proposed GA HDTV system standard.

EXECUTIVE SUMMARY

LAYERED SYSTEM ARCHITECTURE WITH HEADER/DESCRIPTORS

The Grand Alliance HDTV System is a layered digital system architecture that uses headers/descriptors to provide flexible operating characteristics. The layers of the GA HDTV System, and some of their most important capabilities, are:

- the Picture layer provides multiple picture formats and frame rates
- the Compression layer uses MPEG-2 video compression and Dolby AC-3 audio compression
- the Transport layer is a packet format based on MPEG-2 transport, that provides the flexibility to deliver a wide variety of picture, sound and data services
- the Transmission layer is a Vestigial Sideband signal that delivers a data rate of over 18 Mbps in the 6 MHz simulcast channel

While all of the GA HDTV system's layers operate in unison as an effective simulcast system, each layer has also been designed to have outstanding interoperability. The GA HDTV system's layered digital system approach with header/descriptors will create interoperability among a wide variety of consumer electronics, telecommunications, and computing equipment.

PICTURE LAYER

The picture layer consists of raw pixel data, organized as pixels, scan lines and frames. The GA HDTV system provides for multiple formats and frame rates, *all of which will be decoded by any GA HDTV receiver*. This approach allows program producers and application developers to make their own tradeoffs among resolution, frame rate, compression artifacts and interlace artifacts, and to choose the format that is best for their particular use. The formats are:

9NSDOCID: <XP_____2055175A__I_>

Spatial Format (X x Y active pixels)	Temporal rate
1280 x 720 (square pixels)	23.976/24 Hz progressive scan
	29.97/30 Hz progressive scan
	59.94/60 Hz progressive scan
1920 x 1080 (square pixels)	23.976/24 Hz progressive scan
	29.97/30 Hz progressive scan
	59.94/60 Hz interlaced scan

COMPRESSION LAYER

The compression layer transforms the raw video and audio samples into a coded bit stream -- essentially a set of computer instructions and data that are executed by the receiver to recreate the pictures and sound. The compression layer of the Grand Alliance HDTV system:

- uses video compression syntax that conforms to the ISO-MPEG (International Standards Organization — Moving Picture Experts Group) MPEG-2 video data compression draft standard, at a nominal data rate of approximately 18.4 Mbps.
- uses Dolby AC-3 audio compression, at a nominal data rate of 384 kbps.

TRANSPORT LAYER

The transport layer separately packetizes video, audio and auxiliary data and allows their mix to vary dynamically, providing the flexibility needed to innovate new services and new kinds of programming. The transport layer of the Grand Alliance HDTV system:

- uses a packet format that is a subset of the MPEG-2 transport protocol.
- provides basic service capabilities that include video, five-channel surround-sound audio and an auxiliary data capacity.
- offers great flexibility in the mix of video, audio and data services that can be provided to appropriately-featured receivers. It separately packages each type of data (e.g., video, audio, etc.) in its own set of transmission packets. Each packet has a *Packet ID* header that identifies the content of the data stream. This capability enables the creation of new services, ranging from many stereo channels of audio, to broadcast distribution of computer software, to the transmission of very high resolution still images to computers.
- allows the mix of services to be dynamically allocated. This capability will allow rapid burst-mode addressing of receivers. It will also enable broadcasters to send multiple "streams" of video, audio and data programming to their audience, all complementing or enhancing the basic program content. This capability can fundamentally change the nature of television programming, since it enables software to be broadcast to "smart receivers" that can operate in conjunction with the HDTV picture and sound. With this capability, HDTV will likely become a more interactive medium than today's television, enabling new forms of educational and entertainment programming and games.
- provides important extensibility, since a Grand Alliance HDTV receiver will disregard any packet with a PID header that it does not recognize or cannot process. This will eliminate future "backward-compatibility" problems in the installed base of receivers, removing a crucial constraint from the introduction of new services.

ANSMISSION LAYER

.e transmission layer modulates a serial bit stream into a signal that can be transmitted over a 6 Hz analog channel. The transmission layer of the Grand Alliance HDTV system:

- uses a trellis-coded 8-VSB modulation technique to deliver approximately 18.8 Mbps in the 6 MHz terrestrial simulcast channel
- provides a pilot tone that facilitates rapid signal acquisition and increases pull-in range.
- provides a training signal that facilitates channel equalization for removing multipath distortion.
- provides a related 16-VSB modulation technique to deliver two 18.8 Mbps data streams in a 6 MHz cable television channel

UMMARY

The GA HDTV system has the flexible operating characteristics that allow it to provide broad nteroperability needed to form the basis for new and innovative services and applications of HDTV in many industries.

3NSDOCID: <XP_____2055175A_I_>

P. 1 - 17 + 27 - 36 = (27)
Pd: 22-02-94

Chapter V
TRANSPORT SYSTEM

INSDOCID: <XP_____2055175A__I_>

The Grand Alliance Transport System

Table of Contents

BNSDOCID: <XP_____2055175A_1_>

The Grand Alliance Transport System

1. Introduction

This document provides a description of the functionality and format of the Grand Alliance transport system. While tutorial in nature, the document provides sufficient technical detail to serve as the operational specification of the transport layer. Issues related to terrestrial broadcast and cable delivery of the ATV service, in the context of ACATS discussions, are addressed in this document. The authors have attempted to make this a stand-alone document, though the reader would gain additional insight from associated ISO-MPEG documents on this and related topics.

In developing the specification of the Grand Alliance transport layer, we have drawn upon the collective experience of the member companies in developing individual systems, as well as the excellent body of work created by the ISO-MPEG standards process. While any system design requires intelligent tradeoffs to be made, selection of a format based on fixed-length packets has maintained a number of simultaneous goals.

1.1. Program vs. Transport stream multiplexing

In general there are two approaches for multiplexing elementary bit streams from multiple applications on to a single channel. One approach is based on the use of fixed length packets and the other on variable length packetization. Both approaches have been used in the MPEG-2 standard. As illustrated in Fig. 1.1, the video and audio elementary streams in both cases go through an initial stage of PES packetization (discussed in greater depth later), which results in variable length PES packets. The process of generating the transmitted bit streams for the two approaches is shown to involve a difference in processing only at the final multiplexing stage.

INSDOCID: <XP_____2055175A_I_>

Transport System

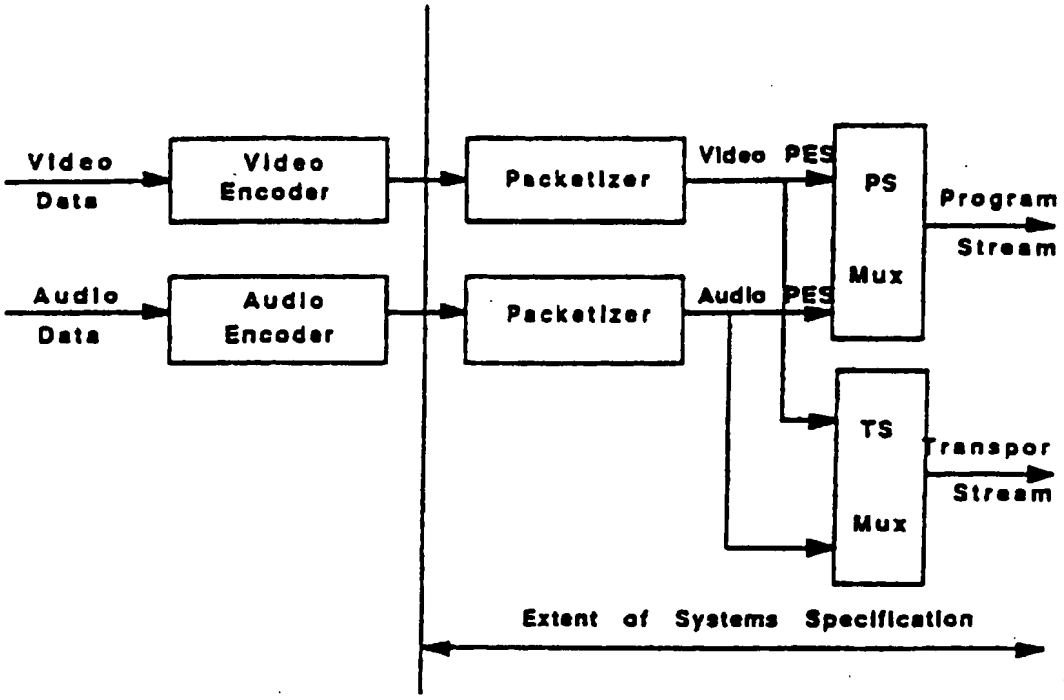


Fig 1.1. Comparison of system level multiplexing approaches.

Fig. 1.2 gives examples of bit streams for the both program and transport stream approaches, in order to clarify their difference. As shown in Fig. 1.2b, in a program stream approach, PES packets from various elementary bit streams are multiplexed by transmitting the bits for the complete PES packets in sequence, thus resulting in a sequence of variable length packets on the channel. (As shown in the diagram, each PES packet is preceded by a PES packet header.) In contrast to this approach, in the transport stream approach selected for the GA system, the PES packets (including the PES headers) are transmitted as the payload of fixed length transport packets. Each transport packet is preceded by a transport header which includes information for bit stream identification. As illustrated in Fig. 1.2a, each PES packet for a particular elementary bit stream occupies a variable number of transport packets, and data from various elementary bit streams are generally interleaved with each other at the transport packet layer, with identification of each elementary bit stream being facilitated by data in the transport headers. New PES packets always start a new transport packet in the GA system, and stuffing bytes are used to fill packets with partial PES data.

3NSDOCID: <XP_____2055175A_I_>

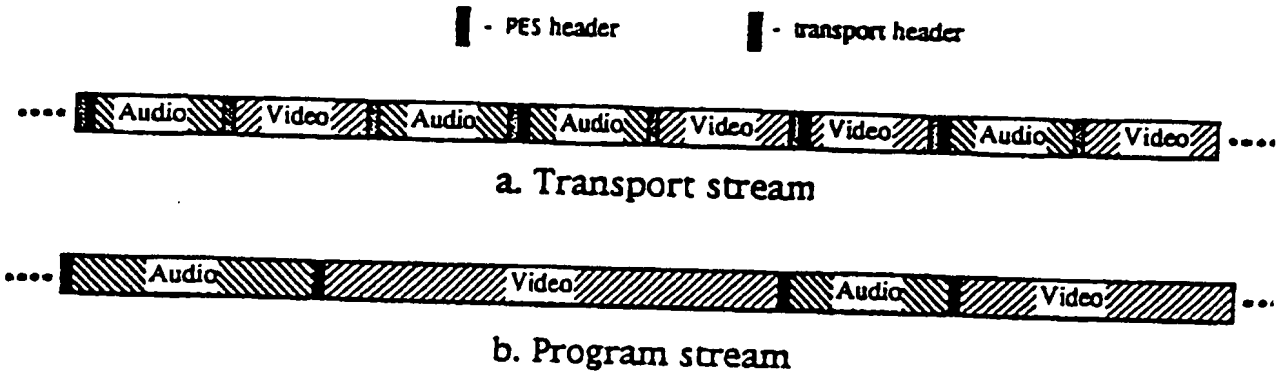


Fig. 1.2. Illustration of bit streams for the two packetization approaches.

The two multiplexing approaches are motivated by completely different application scenarios. Transport streams are defined for environments where errors and data loss events are likely, including storage applications and transmission on noisy channels. Program streams on the other hand are designed for relatively error-free media, e.g. CD-ROMs. Errors or loss of data within PES packets can potentially result in complete loss of synchronization in the decoding process in this case. The definition of program stream approach within MPEG-2 is also motivated by the requirement for compatibility with MPEG-1.

The transport stream approach of MPEG-2 has been found to support most of the functional requirements for the GA system, and hence forms the basis of the GA system definition. As will become clearer from discussions in the following sections, the variable packet-size based program stream approach does not meet these requirements in many aspects. Furthermore compatibility with MPEG-1 systems (which is based on the program stream concept) is not a concern for the GA. Note that the GA system transport system can still identify and carry MPEG-1 video and audio services. In general, the program and transport streams both address the same general layers of protocol functionality and therefore it does not make much sense to attempt to carry a program bit stream within a transport stream or vice-versa. Transcoding between the two formats is however feasible and one could in theory build an interface that connects from a GA bit stream to a program stream decoder. The need for such functionality is not anticipated.

Another point to note is that in other ATV scenarios such as CATV and DBS, defacto standards are being set based on the use of the fixed length packetization approach. The approach for the GA system is consistent with the need for simple interoperability with these scenarios.

INSDOCID: <XP_____2055175A_I_>

1.2. Advantages of the fixed length packetization approach

The fixed length packetization approach offers a great deal of flexibility and some additional advantages when attempting to multiplex data related to several applications on a single bit stream. These are described in some detail in this section.

Dynamic Capacity Allocation:

While digital systems are generally described as flexible, the use of fixed length packets offers complete flexibility to allocate channel capacity among video, audio and auxiliary data services. The use of a packet id (or PID) in the packet header as a means of bit stream identification makes it possible to have a mix of video, audio and auxiliary data which is flexible and which need not be specified in advance. The entire channel capacity can be reallocated in bursts for data delivery. This capability could be used to distribute decryption keys to a large audience of receivers during the seconds preceding a popular pay-per-view program, or download program-related, computer software to a "smart receiver."

Scalability:

The transport format is scalable in the sense that availability of a larger bandwidth may also be exploited by adding more elementary bit streams at the input of the multiplexer, or even multiplexing these elementary bit streams at the second multiplexing stage with the original bit stream. This is a critical feature for network distribution, and also serves interoperability with a cable plant's capability to deliver a higher data rate within a 6 MHz channel.

Extensibility:

Because there will be possibilities for future services that we cannot anticipate today, it is extremely important that the transport architecture provide open-ended extensibility of services. New elementary bit streams could be handled at the transport layer without hardware modifications, by assigning new packet IDs at the transmitter and filtering on these new PIDs in the bit stream at the receiver. Backward compatibility is assured when new bit streams are introduced into the transport system since existing decoders will automatically ignore new PIDs. This capability could possibly be used to compatibly introduce "1000-line progressive formats" or "3D- HDTV" by sending augmentation data along with the normal ATV data.

Robustness:

Another fundamental advantage of the fixed length packetization approach is that the fixed length packet can form the basis for handling errors that occur during transmission. Error correction and

INSDOCID: <XP_____2055175A_1_>

detection processing (which precedes packet demultiplexing in the receiver subsystem) may be synchronized to the packet structure so that one deals at the decoder with units of packets when handling data loss due to transmission impairments. Essentially, after detecting errors during transmission, one recovers the data bit stream from the first good packet. Recovery of synchronization within each application is also aided by the transport packet header information. Without this approach, recovery of synchronization in the bit streams would have been completely dependent on the properties of each elementary bit stream.

Cost Effective Receiver Implementations:

A fixed-length packet based transport system enables simple decoder bit stream demultiplex architectures, suitable for high speed implementations. The decoder does not need detailed knowledge of the multiplexing strategy or the source bit-rate characteristics to extract individual elementary bit streams at the demultiplexer. All the receiver needs to know is the identity of the packet, which is transmitted in each packet header at fixed and known locations in the bit stream. The only important timing information is for bit level and packet level synchronization.

MPEG-2 Compatibility:

The GA transport system is based on the MPEG-2 system specification. While the MPEG-2 system layer has been designed to support many different transmission and storage scenarios, care has been taken by MPEG, as well as the Grand Alliance, to limit the burden of protocol inefficiencies caused by this generality in definition.

An additional advantage of MPEG-2 compatibility is interoperability with other MPEG-2 applications. The MPEG-2 format is likely to be used for a number of other applications, including storage of compressed bit streams, computer networking, and non-HDTV television delivery systems. MPEG-2 transport system compatibility implies that GA transport bit streams may directly be handled in these scenarios (ignoring for the moment the issue of bandwidth and processing speed).

While the GA transport format conforms to the MPEG-2 systems format, it will not exercise all the capabilities defined in the MPEG-2 transport. Therefore, a GA System decoder need not be fully MPEG-2 systems compliant, in that it will not be able to decode any arbitrary MPEG-2 systems bit streams. However, all MPEG-2 decoders should be able to decode the GA bit stream syntax at the transport system level. Documents defining the extent to which the MPEG capabilities are supported in the GA transport have been submitted to the MPEG committee and have contributed to the current working draft of the standard. (See Attachment 1.) MPEG-2 standard features not

INSDOCID: <XP_____2055175A_1_>

supported in the GA specification were constrained[1] if they were deemed to not be applicable to broadcast/cable delivery of ATV.

In the development of the GA transport specification, the intent has never been to limit the design by the scope of the MPEG-2 systems definition. If the MPEG-2 standard is unable to efficiently meet the requirements of the GA system, a deviation from MPEG would be in order. The Advisory Committee would be notified should there be a future deviation from MPEG, with justification for the change.

1.3. Overview of the transport subsystem

Fig. 1.3. illustrates the organization of a GA transmitter-receiver pair and the location of the transport subsystem in the overall system. The transport resides between the application (e.g. audio or video) encoding/decoding function and the transmission subsystems. At its lowest layer, the encoder transport subsystem is responsible for formatting the encoded bits and multiplexing the different components of the program for transmission. At the receiver, it is responsible for recovering the bit streams for the individual application decoders and for the corresponding error signaling. (At a higher layer, multiplexing and demultiplexing of multiple programs within a single bit stream can be achieved with an additional system level multiplexing or demultiplexing stage before/after the modem in the transmitter/receiver.) The transport subsystem also incorporates other higher level functionality related to identification of applications and, as illustrated, synchronization of the receiver. This document will describe these functions in greater detail.

[1] The constraint takes the form of a limitation of functionality. In this instance, certain flags will be permanently configured, and some fields will not appear in the Grand Alliance bitstream. This allows a simpler decoder, as it will not be necessary to handle elements not used for the Grand Alliance application. Often constraints are grouped as a "profile" when acknowledged by the MPEG standards body.

BNSDOCID: <XP_____2055175A__I_>

Transport System

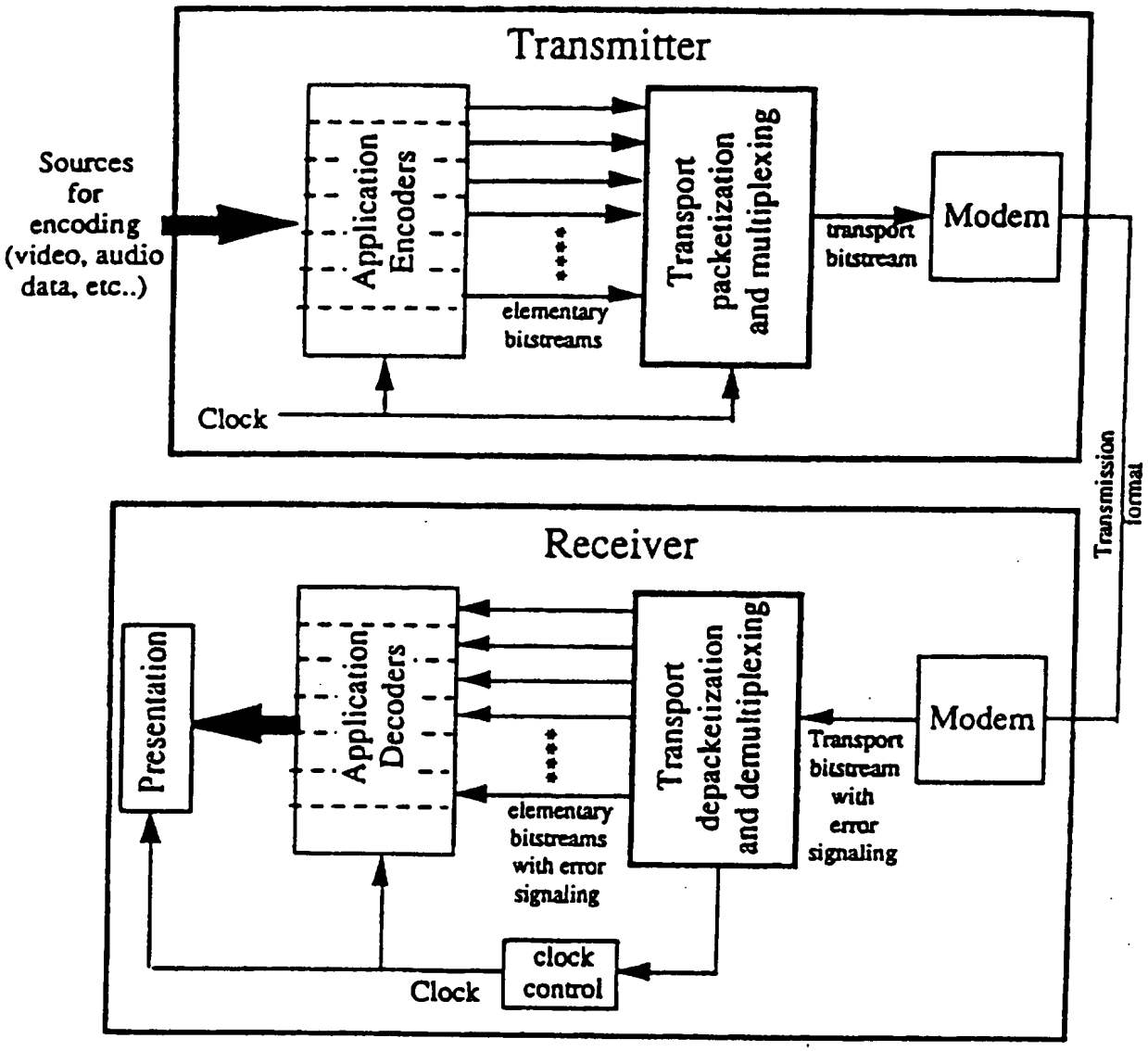


Fig. 1.3. Sample organization of functionality in a transmitter-receiver pair for a single GA program.

As described earlier, the data transport mechanism in the GA system is based on the use of fixed length packets that are identified by headers. Each header identifies a particular application bit stream (also called an *elementary bit stream*) which forms the payload of the packets. Applications supported include video, audio, data, program and system control information, etc.. As indicated earlier, the elementary bit streams for video and audio are themselves be wrapped in a variable length packet structure called the packet elementary stream (PES) before transport processing. The PES layer provides functionality for identification, and synchronization of decoding and

NSDOCID: <XP_____2055175A_I_>

presentation of the individual application. The format and functionality of a PES packet is described in a later section.[2]

Moving one level up in the description of the general organization of the GA bit streams, elementary bit streams sharing a common time base are multiplexed, along with a control data stream, into *programs*. These programs and an overall system control data stream are then asynchronously multiplexed to form a multiplexed *system*. The organization is described in detail in a later section. Note that programs in the GA system are analogous to today's NTSC broadcast channels.

At this level, the GA transport is also quite flexible in two aspects:
1. It permits you to define programs as any combination of elementary bit streams, for example the same elementary bit stream could be present in more than one program (e.g., two bit streams with the same audio), a program could be formed by combining a basic elementary bit stream and a supplementary elementary bit stream (i.e., bit streams for scalable decoders), programs could be tailored for specific needs (e.g., regional selection of language for broadcast of secondary audio), etc....

2. Flexibility at the systems layer allows different programs to be multiplexed into the system as desired, and allows the system to be reconfigured easily when required. The procedure for extraction of programs from within a system is also simple and well defined.

The GA format provides other features that are useful for both normal decoder operation and for the special features required in broadcast and cable applications. These include

1. Decoder synchronization
2. Conditional access
3. Local program insertion, etc....

The elements of these features that are relevant to the standard definition process will be discussed in detail.

The GA bit stream definition directly addresses issues related the storage and playback of programs. Although, this is not directly related to the ATV transmission problem, it is a fundamental requirement for creating programs in advance, storing them and playing them back at the desired time. The programs are stored in the same format in which they are transmitted, i.e., as

[2]Note that the PES layer is not required for all applications. Its use is mandated for both the video and audio in the GA system.

BNSDOCID: <XP_____2055175A_I_>

transport bit streams. The GA bit stream format also has the hooks in it to support the design of consumer digital products based on recording and playback of these bit streams, including the use of the "trick modes" that one is familiar with for current analog VCRs. It should be noted that the issues related to storage and play back of digitally compressed video bit streams are quite different from those that need to be considered for analog systems such as NTSC.

1.4. General bit stream interoperability issues

The question has been raised frequently about the bit stream level interoperability of the GA system. There are two sides to this issue. One is whether the GA transport bit stream can be carried on other communication systems, and the other is the ability of the GA system to carry bit streams generated from other communication systems.

The first aspect of transmitting GA bit streams in different communication systems has been addressed to some extent (e.g., for ATM interoperability) in the design of the protocol, and is described in more detail in later sections. In short, there is nothing that prevents the transmission of a GA bit stream as the payload on a different transmission system. It may be simpler to achieve this functionality in certain systems, e.g., CATV, DBS, ATM, etc.., than in others, e.g., computer networks based on protocols such as FDDI, IEEE 802.6, etc.. Since ATM is expected to form the basis of future broadband communications, it is believed that the issue of bit stream interoperability has been addressed for one of the more important transmission scenarios of the future. This is discussed in more detail in Chapter 9.

The other aspect is of transmitting other, non-GA, bit streams within the GA system. This makes more sense for bit streams linked to TV broadcast applications, e.g., CATV, DBS, etc.., but is also possible for other "private" bit streams. This function is achieved by transmitting these other bit streams as the payload of identifiable transport packets. The only requirement is to have the general nature of these bit streams recognized within the GA system context. Note that there is also a certain minimum system level processing defined by the GA that needs to be implemented to extract all (even private) bit streams. The details are made clearer in the sections that follow. It is also important to remember that the GA system is essentially a broadcast system and hence any private transmissions that may be based on a two way communications protocol will not be directly supported, unless this functionality is provided external to the GA system definition.

NSDOCID: <XP_____2055175A_I_>

2. The packetization approach and functionality

The GA transport bit stream consists of fixed length packets with a fixed and a variable component to the header field as illustrated in Fig. 2.1.

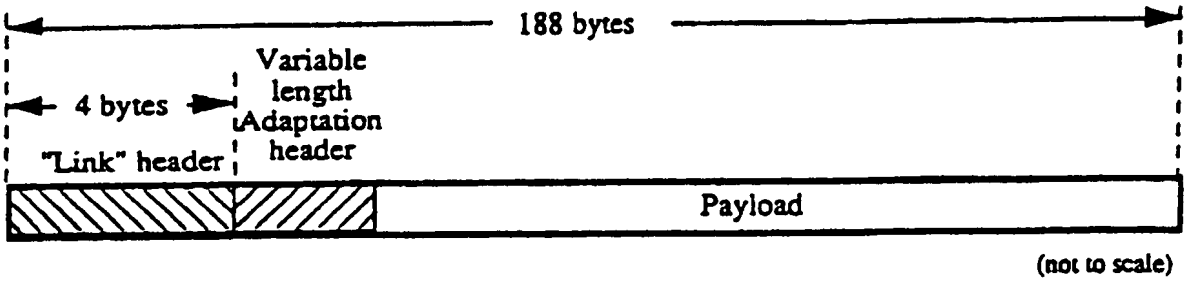


Fig. 2.1. GA Transport packet format

Each packet consists of 188 bytes. The choice of this packet size is motivated by a few factors. The packets need to be large enough so that the overhead due to the transport headers do not become a significant portion of the total data carried. They should not be too large that the probability of packet error becomes significant under standard operating conditions (due to inefficient error correction). It is also desirable to have packet lengths in tune with the block sizes of typical, block oriented, error correction approaches, so that packets may be synchronized to error correction blocks, and the physical layer of the system can aid the packet level synchronization process in the decoder. Another motive for the particular packet length selection is interoperability with the ATM format. The general philosophy is to transmit a single GA transport packet in four ATM cells. There are, in general, several approaches to achieve this functionality. Chapter 9 includes a discussion of some example approaches. If this interface is to be standardized, the issue will be settled outside the scope of the GA/ACATS process.

The contents of each packet and the nature of this data are identified by the packet headers. The packet header structure is layered and may be described as a combination of a fixed length "link" layer and a variable length adaptation layer. Each layer serves a different functionality similar to the link and transport layer functions in the OSI layers of a communications system. This link and adaptation level functionality is directly used for the terrestrial link on which the GA bit stream is transmitted. However these headers could also be completely ignored in a different system (e.g. ATM), in which the GA bit stream may just remain the payload to be carried. In this environment,

INSDOCID: <XP_____2055175A_I_>

the GA bit stream headers would serve more as an identifier for the contents of a data stream rather than as a means for implementing a protocol layer in the overall transmission system.

2.1. The "link" layer

The link layer is implemented using a four byte header field. The format of the header field is described in greater detail in a later section. Some of the important functions that are enabled by the header elements are described here.

2.1.1. Packet synchronization

Packet synchronization is enabled by the sync_byte, which is the first byte in a packet. The sync_byte has the same fixed, pre-assigned, value for all GA bit streams. In some implementations of decoders the packet synchronization function is done at the physical layer of the communication link (which precedes the packet demultiplexing stage), in which case this sync_byte field may be used for verification of packet synchronization function. In other decoder implementations this byte may be used as the primary source of information for establishing packet synchronization. The standard does not specify the details of the approach to be used to implement this function in a decoder but only provides the hooks in the bit stream to facilitate the function.

2.1.2. Packet Identification

As discussed earlier, an important element in the link header is a 13 bit field called the PID or Packet ID. This provides the mechanism for multiplexing and demultiplexing bit streams, by enabling identification of packets belonging to a particular elementary or control bit stream. Since the location of the PID field in the header is always fixed, extraction of the packets corresponding to a particular elementary bit stream is very simply achieved once packet synchronization is established by filtering packets based on PIDs. The fixed packet length makes for simple filter and demultiplexing implementations suitable for high speed transmission systems.

2.1.3. Error handling

Error detection is enabled at the packet layer in the decoder through the use of the continuity_counter field. At the transmitter end, the value in this field cycles from 0 through 15 for all packets with the same PID that carry a data payload (as will be seen later, the GA transport allows you to define packets that have no data payload). At the receiver end, under normal conditions, the reception of packets in a PID stream with a discontinuity in the continuity_counter value indicates that data has been lost in transmission. The transport processor at the decoder then signals the decoder for the particular elementary stream about the loss of data. This signaling approach is not included in the standard.

NSDOCID: <XP_____2055175A_I_>

Because certain information (such as headers, time stamps, and program maps) is very important to the smooth and continuous operation of a system, the GA transport system has a means of increasing the robustness of this information to channel errors by providing a mechanism for the encoder to duplicate packets. Those packets that contain important information will be duplicated at the encoder. At the decoder, the duplicate packets are either used if the original packet was in error or are dropped. Semantics for identifying duplicate packets are described in the description of the continuity_counter.

2.1.4. Conditional Access

The transport format allows for scrambling of data in the packets. Each elementary bit stream in the system can be scrambled independently. The GA standard specifies the descrambling approach to be used but does not specify the descrambling key and how it is obtained at the decoder. The key must be delivered to the decoder within a time interval of its usefulness. There is "private" data capacity at several locations within the GA transport stream where this data might be carried. Two likely locations would be 1) as a separate private stream with it's own PID, or 2) a private field within an adaptation header carried by the PID of the signal being scrambled. The security of the conditional access system is ensured by encrypting the descrambling key when sending it to the receiver, and by updating the key frequently. As mentioned before, the key encryption, transmission, and decryption approaches are not a part of the standard and could differ in different versions of the ATV delivery system. There is no system imposed limit on the number of keys that can be used and the rate at which these may be changed. The only requirement in a receiver to meet the standard is to have an interface from the decryption hardware (e.g., a Smart-card) to the decoder that meets the standardized interface spec. The decryption approach and technology is itself not a part of the standard. For the purposes of testing, to demonstrate feasibility, only the descrambling function will be tested and the encryption keys will probably be made directly available at the decoder. Note that the generalized systems layer definition includes a mechanism for transmitting key information, including the process of identifying bit streams carrying key information and the definition of the tables that enable this function.

Information in the link header of a transport packet describes if the payload in the packet is scrambled and if so, flags the key to be used for descrambling. The header information in a packet is always transmitted in the clear, i.e., unscrambled. The amount of data to be scrambled in a packet is variable depending on the length of the adaptation header. It should be noted that some padding of the adaptation field might be necessary for certain block mode algorithms. Conditional access is discussed in greater detail in a later section.

BNSDOCID: <XP_____2055175A_1_>

Note that the general MPEG-2 transport definition provides the mechanism to scramble at two levels, within the PES packet structure and at the transport layer. Scrambling at the PES packet layer is primarily useful in the program stream (which is not supported in the GA system), where there is no protocol layer similar to the transport to enable this function. In the GA system scrambling will be implemented only at the transport layer.

2.2. The Adaptation layer

The adaptation header in the GA packet is a variable length field. Its presence is flagged in the link level section of the header. The functionality of these headers is basically related to the decoding of the elementary bit stream that is extracted using the link level functions. Some of the functions of this layer that are important to the functioning of the GA system are described here.

2.2.1. Synchronization and timing

Synchronization of the decoding and presentation process for the applications running at a receiver is a particularly important aspect of real time digital data delivery systems such as the GA system. Since received data is expected to be processed at a particular rate (to match the rate at which it is generated and transmitted), loss of synchronization leads to either buffer overflow or underflow at the decoder, and as a consequence, loss of presentation/display synchronization. The problems in dealing with this issue for a digital compressed bit stream are different from those for analog NTSC. In NTSC, information is transmitted for the pictures in a synchronous manner, so that one can derive a clock directly from the picture synch. In a digital compressed system the amount of data generated for each picture is variable (based on the picture coding approach and complexity), and timing cannot be derived directly from the start of picture data. Indeed, there is really no natural concept of synch pulses (that one is familiar with in NTSC) in a digital bit stream.

The solution to this issue in the GA system is to transmit timing information in the adaptation headers of selected packets, to serve as a reference for timing comparison at the decoder. This is done by transmitting a sample of a 27 MHz clock in the program_clock_reference (PCR) field, which indicates the expected time at the completion of the reading of that field from the bit stream at the transport decoder. The phase of the local clock running at the decoder is compared to the PCR value in the bit stream at the instant at which it is obtained, to determine whether the decoding process is synchronized. In general, the PCR from the bit stream does not directly change the phase of the local clock but only serves as an input to adjust the clock rate. Exceptions are during channel change and insertion of local programming. As mentioned earlier, the nominal clock rate in the GA decoder system is 27 MHz. A point to note here is that the standard only specifies the

INSDOCID: <XP_____2055175A_I_>

means of transmitting synchronization information to a receiver but does not specify the implementation of the synch recovery process. Note also that the audio and video sample clocks in the decoder system are locked to the system clock derived from the PCR values. This simplifies the receiver implementation in terms of the number of local oscillators required to drive the complete decoding process, and has other advantages such as rapid synch acquisition.

Details of the format for the PCR are given in a later section. Note that in this implementation the encoder and decoder system clocks are set completely independent of the modem clock. This makes for a clean separation of functionality when implementing the two subsystems, and leads to simpler interfaces. This also makes it simpler to interface GA transmitters and receivers at the transport interface to modems which may be used for transmission on other media such as CATV, DBS, computer networks, etc..

2.2.2. Random entry into the compressed bit stream

Random entry into the application bit streams such as video and audio is necessary to support functions such as program tuning and program switching. Random entry into an application is possible only if the coding for the elementary bit stream for the application supports this functionality directly. For example, a GA video bit stream supports random entry through the concept of Intra (or I-) frames that are coded without any prediction, and which can therefore be decoded without any prior information. The beginning of the video sequence header information preceding data for an I-frame could serve as a random entry point into a video elementary bit stream. In general, random entry points should also coincide with the start of PES packets where they are used, e.g., for video and audio. The support for random entry at the transport layer comes from a flag in the adaptation header of the packet that indicates whether the packet contains a random access point for the elementary bit stream. In addition, the data payload of packets that are random access points also start with the data that forms the random access points into the elementary bit stream itself. This approach allows the discarding of packets directly at the transport layer when switching channels and searching for a resynchronization point in the transport bit stream, and also simplifies the search for the random access point in the elementary bit stream once transport level resynchronization is achieved.

A general objective is to have random entry points into the programs as frequently as possible, to enable rapid channel switching.

INSDOCID: <XP_____2055175A_I_>

2.2.3. Local program insertion

This functionality is important for inserting local programming, e.g., commercials, into a bit stream at a broadcast headend. In general, there are only certain fixed points in the elementary bit streams at which program insertion is allowed. The local insertion point has to be a random entry point but not all random entry points are suitable for program insertion. For example, for GA video, in addition to being a random entry point, the VBV_delay (video buffer verifier delay) needs to be at a certain system defined level to permit local program insertion.[3] This is a requirement to control the memory needed at the decoder for buffering data and to prevent buffer overflow or underflow. Local program insertion also always takes place at the transport packet layer, i.e., the data stream splice points are packet aligned. Implementation of the program insertion process by the broadcaster is aided by the use of a splice_countdown field in the adaptation header that indicates ahead of time the number of packets to countdown until the packet after which splicing and local program insertion is possible. The insertion of local programming usually results in a discontinuity in the values of the PCR received at the decoder. Since this change in PCR is completely unexpected (change in PCR values are usually only expected during program change), the decoder clock could be thrown completely out of synchronization. To prevent this from happening, information is transmitted in the adaptation header of the first packet after the splicing point to notify the decoder of the change of PCR values (so that it can change the clock phase directly instead of attempting to modify the clock rate). In addition there are constraints on 1) the length of the bit stream that is to be spliced in, to assure that the buffer occupancies at the decoder both with and without the splice would be consistent, and 2) the initial VBV value assumed when encoding the bit stream to be spliced in, in order to prevent decoder buffer underflow or overflow.

The details of the syntax elements that support splicing and local program insertion are described in the chapter on the transport format. More specifics of the particular implementation for the GA system will be described in a separate section.

In addition to the functions described above, the adaptation header includes capabilities for extension of the header, for supporting new functionality, and also for defining data that is private and whose format and meaning are not defined in the public domain. These elements may be useful in extending the GA transport beyond the current range of expectations for usage.

---

[3]The VBV_delay information is computed and transmitted as a part of the header data for a picture in the compressed video bit stream. It defines how full the decoder video buffer should be just before the bits for the current picture are extracted from the buffer, if the decoder and encoder processes are synchronized.

3NSDOCID: <XP_____2055175A_I_>

3. Higher Level Multiplexing functionality

As described earlier, the overall multiplexing approach can be described as a combination of multiplexing at two different layers. In the first layer one forms program transport streams by multiplexing one or more elementary bit streams at the transport layer, and in the second layer the program transport streams are combined (using asynchronous packet multiplexing) to form the overall system. The functional layer in the system that contains both this program and system level information that is going to be described is called the PSI or Program Specific Information.

3.1. Single Program Transport Multiplex

A GA program transport bit stream[4] is formed by multiplexing individual transport packetized elementary bit streams (with or without PES packetization) sharing a common time-base, and a control bit stream that describes the program. Each elementary bit stream, and the control bit stream (also called the elementary stream map in Fig. 3.1), are identified by their unique PIDs in the link header field. The organization of this multiplex function is illustrated in Fig. 3.1. The control bit stream contains the program_map_table that describes the elementary stream map. The program_map_table includes information about the PIDs of the transport streams that make up the program, the identification of the applications that are being transmitted on these bit streams, the relationship between these bit streams, etc.. The details of the program_map_table syntax and the functionality of each syntax element are given in a later section. The identification of a bit stream carrying a program_map_table is done at the system layers to be described next.

The transport syntax allows a program to be comprised of a large number of elementary bit streams, with no restriction on the types of applications required within a program. A program transport stream does not need to contain compressed video or audio bit streams, or, for example, it could contain multiple audio bit streams for a given video bit stream. The data applications that can be carried are flexible, the only constraint being that there should be an appropriate stream_type ID assignment for recognition of the application corresponding to the bit stream in a GA decoder. The list of application types that will be a supported in the initial GA system are given in the section on services supported by the GA system. Note that the initial selection of applications does not limit the future. (Indeed, it is quite impossible for one to anticipate all possible future applications!)

---

[4]The terminology can be confusing. A program is analogous to a channel in NTSC. A program stream refers to a particular bitstream format, described in the introduction, that is not being used in the GA system. A program transport stream is a term used to describe a transport bitstream that has been generated for a program.

BNSDOCID: <XP_____2055175A_I_>

Note that many of the link level functions are carried out independently, without program level coordination, for the different elementary bit streams that make up a program. This includes functions such as PID manipulation, bit stream filtering, scrambling and descrambling, definition of random entry packets, etc.. The coordination between the elements of a program is primarily controlled at the presentation (display) stage based on the use of the common time base. This common time base is set up by the fact that all elementary bit streams in a program derive timing information from a single clock, the information for which is transmitted via the PCR on one of the elementary bit streams that constitute the program. The data for timing of presentation is present in the elementary bit streams for individual applications.

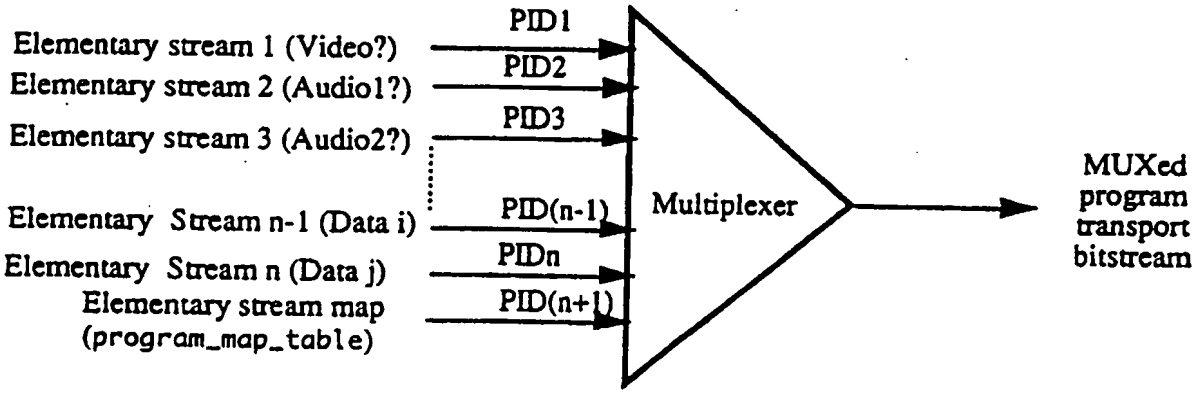


Fig. 3.1. Illustration of the multiplex function to form a program transport stream.

Although there is no restriction on the PID values that can be assigned within a program transport multiplex, in light of the fact that PIDs need to be unique at a system level, a standardized PID assignment approach should be considered for the GA system. A suggestion is to use the LSB bits in the PIDs to identify the stream type.

3.2. System Multiplex

The system multiplex is the process of multiplexing different program transport streams. In addition to the transport bit streams (with the corresponding PIDs) that define the individual programs, a system level control bit stream with PID = 0 is defined. This bit stream carries the program_association_table that maps program identities to their program transport streams. The program identity is represented by a number in the program_association_table. A program corresponds to what is traditionally called a channel, e.g., HBO™, ESPN™, etc.. The map indicates the PID of the bit stream containing the program_map_table for a program. Thus, the process of identifying a program and its contents takes place in two stages: first one uses the program_association_table in the PID = 0 bit stream to identify the PID of the bit stream carrying the program_map_table for the

NSDOCID: <XP_____2055175A_I_>

program, in the next stage one obtains the PIDs of the elementary bit streams that make up the program from the appropriate program_map_table. Once this step is completed the filters at a demultiplexer can be set to receive the transport bit streams that correspond to the program of interest.

The system layer of multiplexing is illustrated in Fig. 3.2. Note that during the process of system level multiplexing, there is the possibility of PIDs on different program streams being identical at the input. This poses a problem since PIDs for different bit streams need to be unique. A solution to this problem lies at the multiplexing stage, where some of the PIDs could be modified just before the multiplex operation. The changes have to be recorded in both the program_association_table and the program_map_table. Hardware implementation of the PID reassignment function in real time is helped by the fact that this process is synchronous at the packet clock rate. The other approach, of course, is to make sure up front that the PIDs being used in the programs that make up the system are unique. This is not always possible with stored bit streams.

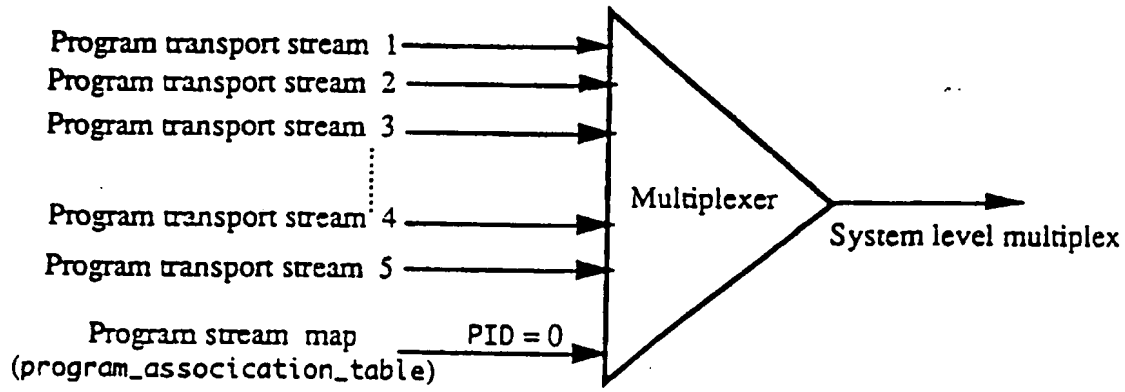


Fig. 3.2. Illustration of the multiplex function to form the system level bit stream.

Note that the architecture of the GA bit stream is scalable. Multiple system level bit streams can be multiplexed together on a higher bandwidth channel by extracting the program_association_tables from each system multiplexed bit stream and reconstructing a new PID = 0 bit stream. Note again that PIDs may have to be reassigned in this case.

Note also that in all descriptions of the higher level multiplexing functionality no mention is made of the functioning of the multiplexer and multiplexing policy that should be used. This function is not a part of the standard and is up to individual designers. Because its basic function is one of filtering, the transport demultiplexer will function on any GA bit stream regardless of the multiplexing algorithm used.

BNSDOCID: <XP_____2055175A_I_>

Fig 3.3 illustrates the entire process of extracting elementary bit streams for a program at a receiver. It also serves as one possible implementation approach (although not the most efficient! In practice the same demultiplexer hardware could be used to extract both the program_association_table and the program_map_table control bitstreams.). This also represents the minimum functionality required at the transport layer to extract any application bit stream (including those that may be private).

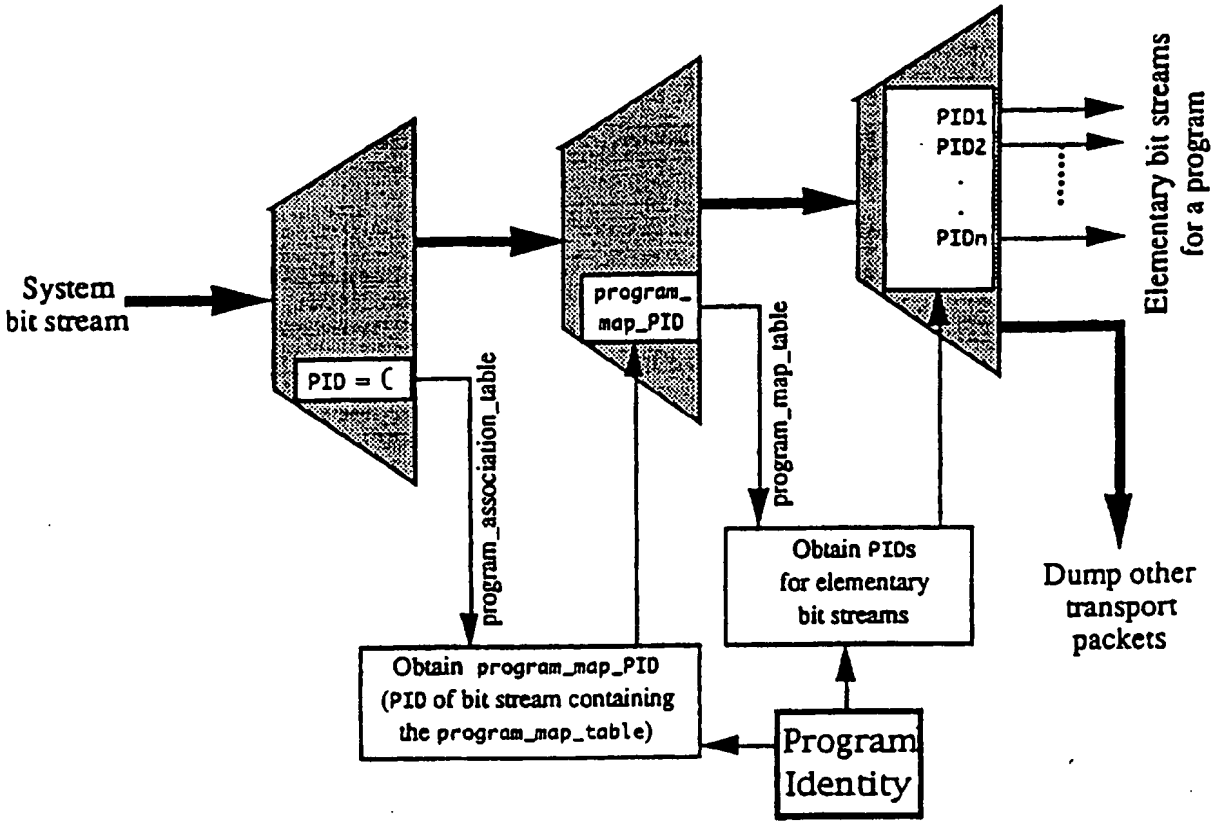


Fig 3.3. Illustration of transport demultiplexing process for a program.

Note that once the packets are obtained for each elementary bit stream in the program, further processing stages of obtaining the random entry points for each component bit stream, decoder system clock synchronization, presentation (or decoding) synchronization, etc.., need to take place before the receiver decoding process reaches normal operating conditions for receiving a program.

It is important to clarify here that the layered approach to defining the multiplexing function does not necessarily imply that program and system multiplexing should always be implemented in

INSDOCID: <XP_____2055175A__I_>

separate stages. A hardware implementation that includes both the program and system level multiplexing within a single multiplexer stage is allowed, as long as the multiplexed output bit stream has the correct properties as defined in this document.

3NSDOCID: <XP_____2055175A_I_>

4. Features and Services Supported by the GA ATV System

As we have demonstrated, the GA Transport architecture has been designed to be maximally flexible and is capable of supporting a vast number of services through its system multiplex. The transport, however, is not the only means by which information can be delivered in the ATV system. The GA video syntax also provides means for delivery of predefined information.

Although this is a global issue for the GA ATV system, it is appropriate to review the services supported in the context of the transport specification, since this sub-system has ultimate responsibility for the multiplexing of supported services. In this chapter, we will review a number of functions identified by ATSC document T3/186 and the SMPTE headers/descriptors group and describe how they are carried within the ATV system. In some cases, these functions are a matter to be communicated between the studio that sources the program and the final encoder that transmits the bit stream. While the final transmitted bit stream does not necessarily need to carry this information to the receiver, it is important to see the capabilities of the ATV system to carry this information in a network distribution capacity.

4.1. Features Supported within the Grand Alliance Video Syntax

Pan & Scan:
Pan & scan information is supported within the GA video syntax. This information is transmitted as an extension within the picture layer syntax. The pan & scan extension allows decoders to define a rectangular region which may be panned around the entire coded image. This facility could be used to identify a 4:3 aspect ratio window within a 16:9 coded image.

Field/Frame Rate and Film Pull-down:
The GA video syntax provides means for transmitting the frame rate of the coded bit stream. This allows the encoder to maximize coding efficiency by not transmitting redundant fields, and signals the decoder the proper order for displaying the decoded pictures. The GA syntax supports frame rates of 23.976, 24, 29.97, 30, 59.94 and 60 Hz as well as an extension for future capabilities. The frame rate syntax is found within the video sequence layer.

Picture Structure Information:
This information details the sampling structure used in the coded image, including samples per line, lines per frame, and scanning format (interlace or progressive). This information is supported by the video syntax and is found within the sequence layer.

3NSDOCID: <XP_____2055175A_I_>

Picture Aspect ratio:
The video syntax provides a field for sample aspect ratio within the sequence layer. This information combined with the picture structure fields, allows the picture aspect ratio to be determined.

Color field Identification:
This information is supported by the GA video syntax and helps the decoder re-encode the image to an NTSC compatible output with reduced NTSC artifacts.

Colorimetry
Information on the colorimetry characteristics of the video to be encoded are supported by syntax in the video sequence layer. This includes description of the color primaries, transfer characteristics and the color matrix coefficients.

4.2. Features Supported as Multiplexed Services within the Grand Alliance Transport System

As mentioned earlier, the GA transport scheme provides great flexibility for multiplexing a variety of services in addition to video and audio elementary streams. Several possible services that could be supported under this transport definition are summarized below.

Audio Compression Types and Language Identification:
The transport layer syntax defines a program map which permits identification of individual audio services by their compression algorithm as well as signaling the presence of a secondary language channel that can be selected by the viewer.

Program Information
This service could be provided to the decoder as an ancillary data service with its own PID. This could take the form of a TV guide that is personalized by the service provider. The information would require only a low refresh rate that would not consume a significant amount of the channel bandwidth.

3NSDOCID: <XP_____2055175A_I_>

Other Program-related Information

There is a large body of program-related information that could be identified for use at the decoder. MPEG-2 systems syntax, on which the GA system is based, currently supports copyright notification, but has not defined a separate capability for program classification data. This and other program-related information could be addressed as private data.

4.3. Support of Closed Captioning

The Grand Alliance has been participating in dialogue with the EIA committee on ATV Closed Caption Standards to ensure proper support of this important feature. The standards group reviewed a number of important requirements for carriage of the closed caption service in a digitally compressed ATV system. In response to those stated needs, the Grand Alliance has indicated that closed captioning would be carried as user data within the video picture layer.

Support of closed caption in this manner, allows a fixed amount of channel capacity to be dedicated to the service, while maintaining absolute synchronization with each ATV frame. Carriage of this data as a separate service would require a separate synchronization mechanism as exists within the audio decoder. This synchronization was an essential requirement stated by the standards group. Additionally, this mechanism allows for relatively easy editing of the closed caption data downstream. It is expected that the closed caption data will require a dedicated allocation of 9600 bits per second.

Our work with this standards body will continue as we progress to final specifications, and our work will be updated to the Advisory Committee.

4.4. Features not Anticipated to be Transmitted by the Grand Alliance System to the Consumer Receiver

Telecine Source Identification:

Source identification information that could inform the encoder whether the video was originally film or video could assist the detection of redundant fields and thereby improve coding efficiency. There is no syntax provided in the video bit stream to carry this external information, however it could be multiplexed in at the systems level as an ancillary data service.

Image Processing History:

It is projected that future encoders could take advantage of knowledge of any algorithms that have been applied to the image sequence. This is knowledge that would need to be passed from the

BNSDOCID: <XP_____2055175A_I_>

studio to the encoder, and would not be sent to the decoder. It could be carried in an ancillary data stream in a wider bandwidth network distribution system.

Scene Change:

Automatic scene change detection algorithms are used in some encoders to improve coding efficiency. There has not been an interface specified for this information to be transmitted to the encoder, but it would aid in the coding algorithm. Such scene change information, if supported by a production facility, could provide useful information to the video encoder at both the compression and transport levels and we look forward to working with standards bodies to make this effective.

Conditional Access Identification:

Implementation of Conditional Access systems is supported by the transport syntax, with bits defined in the packet header. Delivering information about the conditional access information, including key information is an issue that would be addressed as private data in the GA syntax. This issue is covered further in a separate chapter on Conditional Access.

Universal Identifier:

Definition of Universal Identifier information has not yet been finalized by SMPTE. The issue of the registration descriptor is addressed more completely in chapter 9.

INSDOCID: <XP_____2055175A_1>

5. The Transport format and protocol

This chapter defines the syntax elements for the transport layer of the GA bit stream. All syntax elements need to be recognized at some level in a GA receiver. Most trigger a response in the transport decoder. A few are present for interoperability with MPEG-2.

5.1. Link level headers

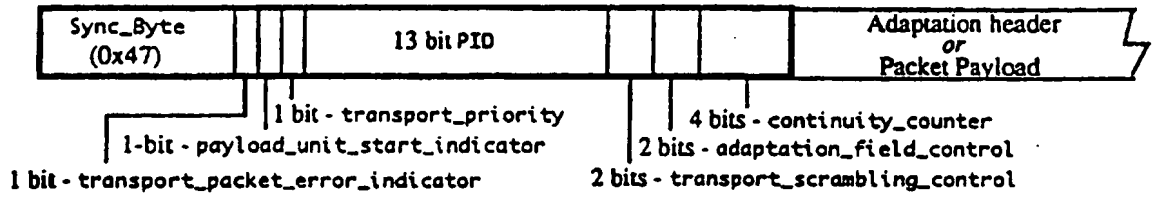


Fig. 5.1. Link header format for the GA system transport packet.

Fig. 5.1. shows the link layer headers with the functionality assigned to each bit. Some of these have been discussed earlier. The table below spells out these functions in detail. These general functions may not all be used on the broadcast channel, but are useful for transmitting the same bit stream over other links, including cable links, computer networks, etc.. In short, these provide interoperability features.

field	General function	GA usage
Sync_byte Value - 0x47	Packet synchronization.	As defined. See section 2.1.1.
transport_packet_error_ indicator	Indicates if packet is erroneous 0 - no error 1 - erroneous packet	Can be used for error signaling from modem to transport demultiplexer. If this bit is set the payload is not supposed to be used.
payload_unit_start_ indicator	Indicates if a PES packet header or the start of a table containing program specific information (PSI) is present in the payload in this packet. The PES packet header always begins the payload of the packet. The starting byte of the PSI table in the packet is indicated using a pointer field to be descibed later. 0 - no PES header or start of PSI table present 1 - PES header present	As defined for resynch into the transport stream.

NSDOCID: <XP_____2055175A_I_>

transport_priority	Priority indicator at input to transmission channels/networks which support prioritization. 0 - lower priority 1 - higher priority	The GA transmission system is currently not expected to support prioritization. If it does, this bit will be set during transport packetization process, to route packets to the transmission path with the appropriate priority.
PID	Packet Identifier for mux/demux.	As defined. See section 2.1.2.
transport_scrambling_control	Indicates the descrambling key to use for the packet 00 - not scrambled others - user defined.	00 - not scrambled 10 - "even" key 11 - "odd" key 01 - reserved See section 2.1.4.
adaptation_field_control	Indicates if an adaptation field follows 00 - reserved 01 - no adaptation field, payload only 10 - adaptation field only, no payload 11 - adaptation field followed by payload	As defined.
continuity_counter	Increments by one for each packet with a given PID and transport priority. Used at the decoder to detect lost packets. Not incremented for packets with adaptation field of 00 or 10. If two consecutive transport packets of the same PID have the same continuity_counter value and the adaptation_field_control equals '01' or '11', the two transport packets shall be considered duplicate.	As defined. See section 2.1.3.

5.2. Adaptation level headers

The presence of the adaptation header field is signaled in the adaptation_field_control of the link layer as described before. The adaptation header itself consists of information useful for higher level decoding functions. The header format is based on the use of flags to indicate the presence of the particular extensions to the field.

The header starts with a fixed length component that is always present (if an adaptation header is transmitted). The format is shown in Fig. 5.2.

3NSDOCID: <XP_____2055175A_I_>

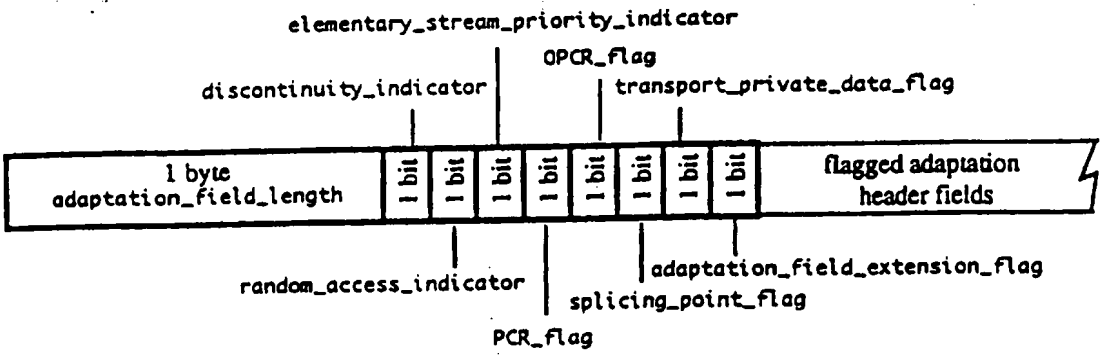


Fig. 5.2. Format for the fixed length component of the adaptation header.

The adaptation_field_length specifies the number of bytes that follow it in the adaptation header. The adaptation header could include stuffing bytes after the last adaptation header component field. (Stuffing bytes have a value of 0xff and are not interpreted at the decoder.) In this case the adaptation_field_length also reflects the stuffing bytes. The value in the adaptation_field_length field can also be used by the decoder to skip over the entire adaptation header, and to directly advance to the data payload in the packet if desired.

The presence of additional adaptation header fields is indicated by the state of the last five single bit flags shown in Fig. 5.2. (with a value of '1' indicating that a particular field is present). The three flags at the beginning do not result in extensions to the adaptation header and are described in the table below.

field	General function	GA usage
discontinuity_indicator	Indicates if there is a discontinuity in the PCR values that will be received from this packet onwards. This happens when bit streams are spliced. This flag should be used at the receiver to change the phase of the local clock.	As defined for local program insertion. See section 2.2.3.
random_access_indicator	Indicates that the packet contains data that can serve as random access point into the bit stream. As an example, these can correspond to the start of sequence header information in the GA video bit stream.	As defined. See section 2.2.2.
elementary_stream_ priority_indicator	Logical indication of priority of the data being transmitted in the packet.	Not used. If set it is ignored at a GA decoder.

As mentioned earlier, the other components of the adaptation header appear based on the state of the flags shown in Fig. 5. The order in which these components appear in the bit stream is the same as the order of the flags. Based on the type of adaptation header information being

INSDOCID: <XP_____2055175A_1_>

conveyed, the data in these fields may be either fixed length or variable length. These fields are described in detail next.

5.2.1. The PCR and OPCR fields

The use of the PCR has been described in detail in section 2.2.1. This section deals mainly with the format for transmission.

Overall Format

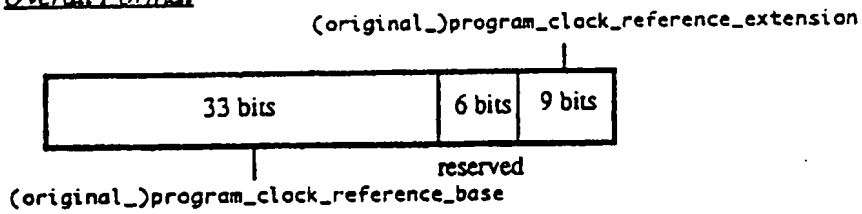


Fig. 5.3. The (O)PCR header format.

Functionality

field	General function	GA usage
PCR	Indicates intended time of arrival of last byte of the program_clock_reference_extension at target decoder. Used for synchronization of the system decoding process. This field can be modified during the transmission process.	As defined. The PCR will be transmitted at least once every 100 milliseconds.
OPCR	Indicates intended time of arrival of last byte of the original_program_clock_reference_extension at target decoder for a single program. This field is not modified during transmission.	May be used for recording and playback of single programs. Not used in the GA receiver in the decoding process.

Functional details of the (O)PCR format

The total PCR value is based on the state of a 27 MHz clock. The 9 bit extension field cycles from 0 to 299 at 27 MHz, at which point the value in the 33 bit field is incremented by one. (This results in the 33 bit field being compatible with the 33 bit field that are used for the 90 KHz clock of MPEG-1. Backward compatibility was a concern in the MPEG-2 system design.) The cycle time of the PCR value is approximately 26 hours.

5.2.2. The transport_private_data and adaptation_field_extension fields

Format

3NSDOCID: <XP_____2055175A_I_>

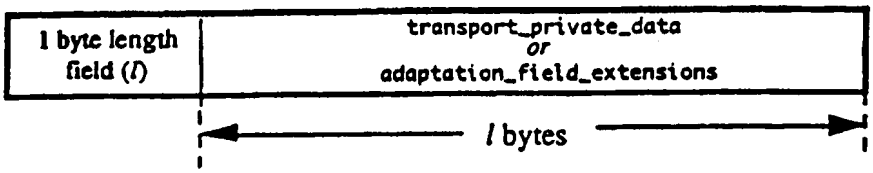


Fig. 5.4. The transport_private_data and adaptation_field_extension header format.

Functionality

field	General function	GA usage
transport_private_data	For private data not recognizable by the general MPEG decoders. Meant for short bursts of control information.	To be decided
adaptation_header_ extensions	For future extensions of the adaptation header which may have not been thought of yet.	Not used currently.

5.2.3. The splice_countdown field

This field is useful for local program insertion as described in section 2.2.3.

Format: This is a one byte field that is present if the splicing_point_flag is set.

Functionality

General Function	GA Usage
Indicates number of packets present in the bit stream, with the same PID as current packet, until a splicing point packet. The splicing point packet is defined as the packet containing a point in the elementary bit stream from which point onwards data can be removed and replaced by another bit stream, so that the resulting transport bit stream is valid according to MPEG-2 rules. Transmitted as a 2s-complement value.	Used for supporting insertion of local programming.

5.3. PSIs and the pointer_field

As mentioned in chapter 3, the program_association_table and the program_map_tables that describe the organization of a multiplexed GA bit stream are a part of the PSI layer of the GA system. PSI tables, in general, are transmitted in the appropriate bit stream sequentially, without any gap between the tables. This implies that tables do not necessarily start at the beginning of a transport packet. This also implies that in order to decode specific tables, there needs to be some indication of where these begin in the bit stream. This functionality is achieved with the pointer_field. The pointer_field, if present, is the byte of the payload of a packet (after the link and adaptation headers). The pointer_field is present in the packet if a PSI table begins in the packet, an event which is signaled at the link level, by setting the payload_unit_start_indicator (described in section 5.1) to '1'.

INSDOCID: <XP_____2055175A_I_>

The pointer_field indicates the number of bytes that follow it before the start of a PSI table. As an example a pointer_field value of 0x00 indicates that a new PSI table begins immediately following it.

5.4. The program_association_table

As discussed in section 3.2, the program_association_table is transmitted as the payload of the bit stream with PID = 0 and describes how program numbers associated with programs, (e.g., HBO™, ESPN™, etc.) map on to bit streams containing the program_map_tables for these programs. This section discusses the syntax of the table in some depth. The program_association_table may be transmitted as multiple program_association_segments with each segment having a maximum length of 1024 bytes. The transport decoder can extract individual table segments from the bit stream in whatever order it desires. As shown in Fig. 5.5, each table segment has a fixed length 10 byte header component for table segment identification, a variable length component that depends on the number of entries contained, and a 4 byte CRC-32 field.

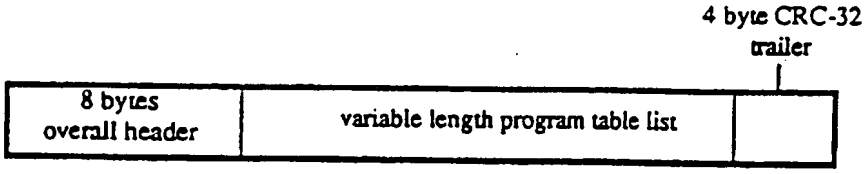


Fig. 5.5. High level overall picture of the program_association_segment.

i. The fixed length overall header component is shown below in Fig. 5.6 and is described in the table below.

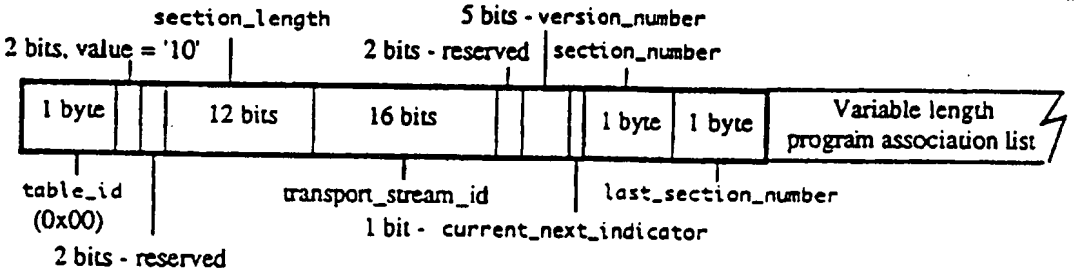


Fig. 5.6. Fixed length header component of the program_association_table.

field	General function	GA usage
table_id	Indicates the nature of the table. 0x00 indicates a program_association_table	As defined.

3NSDOCID: <XP_____2055175A_1_>

section_length	Length of the section of the program_association_table. This length includes all bytes following this field, up to and including the CRC. The two most significant bits of this field are set to 0, i.e., maximum value is 1024. This field allows the transport decoder to skip sections when reading from the bit stream if desired.	As defined.
transport_stream_id	Identication of a particular multiplex from several in the network.	*Should correspond to a channel number for the terrestrial application.*
version_number	Incremented each time there is a change in the program_association_ table being transmitted.	As defined
current_next_indicator	Value of 1 indicates that the map is currently valid. Value of 0 indicates that the map is not currently being used and will be used next.	As defined
section_number	Identifies the particular section being transmitted.	As defined
last_section_number	Section_number for the last section in the program_association_ table. Needed to confirm when an entire program_association_table has been received at the decoder.	As defined

The value of the reserved bits is undefined and the GA system should not interpret these. On the other hand the 2 bit '10' value following the table_id needs to be received correctly.

ii. The variable length component of the table consists of program_count number of fixed length entries corresponding to each program, and stuffing_bytes (to make up the program_association_segment_length). The format for each fixed length entry is shown in Fig. 5.7.

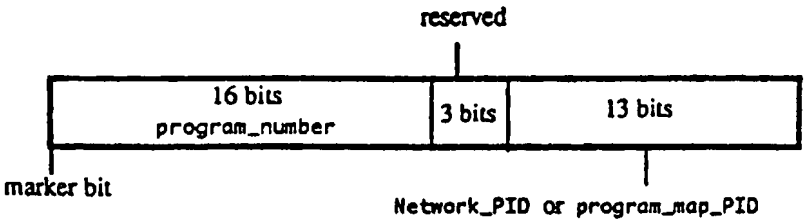


Fig. 5.7. Format of each entry in the program_association_table.

The program identity '0' is reserved for the network_PID, i.e., the PID of the bit stream carrying information about the configuration of the overall system. The nature of this bit stream is yet to be defined for the GA system. The format for this bit stream is completely open since it is meant to be a private bit stream. For all other program identities, the program_map_PID is the PID of the bit stream containing the program_map_table for the particular program.

INSDOCID: <XP_____2055175A__I_>

iii. The program_association_table ends with a four byte CRC field that contains results of a CRC done over the entire program map segment, starting at the segment_start_code_prefix. The CRC is based on the polynomial $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$.

5.5. The program_map_table

As discussed in the previous section, the program_map_table is transmitted as the payload of the bit stream with PID = program_map_PID (as indicated in the program_association_table). The program_map_table carries information about the applications that make up programs. Basically each program_map_table is transmitted as a single TS_program_map_section. The format for a TS_program_map_section can be described as a combination of an overall header field, fields that describe each program within the table and a trailer CRC field, as shown in Fig. 5.8. The CRC used is the same as for the program_association_table. In general, each program_map_PID may contain more than one TS_program_map_section, each describing a different program.

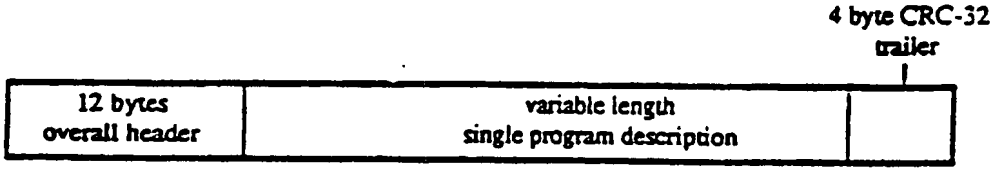


Fig. 5.8. High level overall view of the TS_program_map_section.

5.5.1. The overall TS_program_map_segment header format

The header format for a TS_program_map_section is shown in Fig. 5.9. The format of the first 8 bytes is the same and has similar functionality as that for the program_association_table. The similarity in format is intended to facilitate simple software decoding of the headers. Note that the table_id for TS_program_map_section is different from that for the program_association_table.

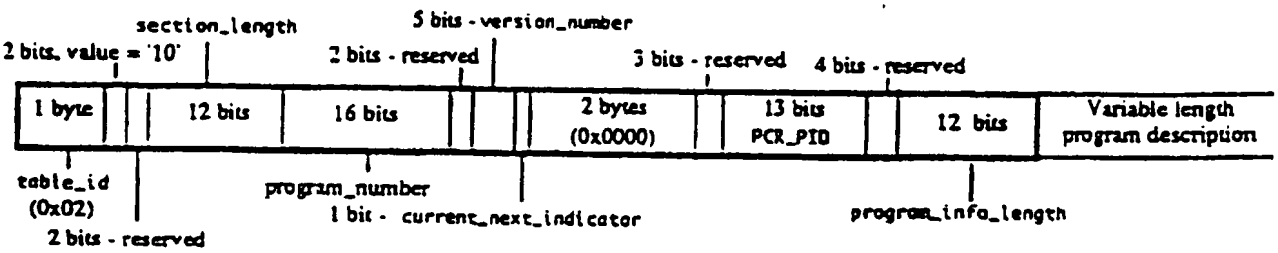


Fig. 5.9. Fixed length header for the TS_program_map_section.

The two bytes that were used to identify the transport_stream_id in the association table are now used to identify the program_number of the program whose description follows. The section identification functions are not required for this table since the description of each program is defined to have to

3NSDOCID: <XP_____2055175A_I_>

fit into one section. Hence these fields are set to '0' as shown in the figure. The other common header fields between the program_association_table and the TS_program_map_section have functionality as desecribed in the table following Fig. 5.8. The additional header fields in a TS_program_map_section are the 13 bit PCR_PID field that identifies the PID of the particular packetized elementary bit stream in the program that contains the PCR values for the program, and the program_info_length field, that indicates the number of bytes of program_descriptors that follow this header field.

The overall program description that follows the header described above consists of the optional, variable length, program_descriptor field (whose length was indicated by the program_info_length field shown in Fig. 5.9), followed by a descriptions of each of the individual elementary bit streams that make up the program, i.e., there are one or more elementary bit stream descriptions for each TS_program_map_section.

5.5.2 An elementary stream description

Each elementary stream description for has a 5 byte fixed length component and a variable length elementary_stream_descriptor component as shown in Fig. 5.10.

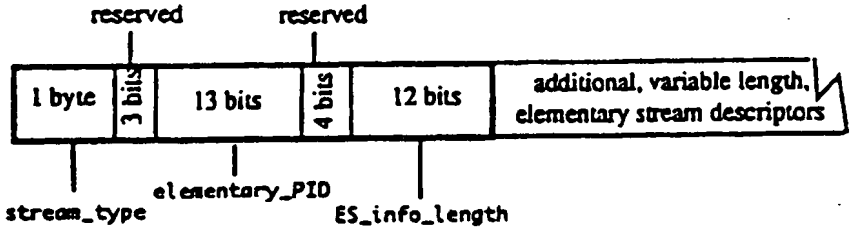


Fig. 5.10. The fixed length component of the elementary stream description.

The functionality of the different fields is as follows.

field	General function	GA usage
stream_type	Indicates the application being considered in this elementary stream.	??.
elementary_pid	Indicates the PID of the transport bit stream containing the elementary bit stream.	As defined
ES_info_length	Indicates the length of a variable length elementary_stream_descriptor field that immediately follows.	As defined.

As before reserved bits are ignored.

NSDOCID: <XP_____2055175A__I_>

5.6. Descriptors

Descriptors are transmitted in the program_descriptor and the elementary_stream_descriptor fields to describe certain charaacterisitics of the program or the elementary bit stream. In general, each program_descriptor and the elementary_stream_descriptor can consist of number of individual descriptor field elements transmitted sequentially.

Two factors need to be considered in order to use descriptors. In the first place, there has to be an mechanism for indicating the presence of the descriptors. In the PSI tables that have been described, this functionality is achieved by the length field that preceeds the descriptor, with a value of zero indicating that no descriptor are present. A second function is the identification of the descriptor itself. This is achieved within the descriptor header itself, which consists of a one byte descriptor_tag field followed by a one byte descriptor_length field that specificies the number of bytes in the descriptor following the descriptor_length field. The set of valid descriptor_tags in the GA systems is the same as that defined for MPEG-2. The table of tags and the format for each descriptor are not described in this document.

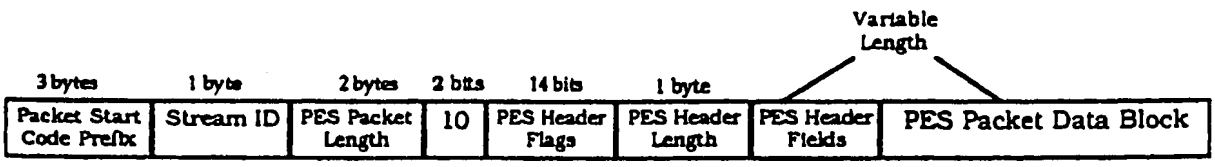
3NSDOCID: <XP_____2055175A_I_>

6. The PES packet format

As described before, some elementary bit streams, including the GA video and compressed audio, will go through a PES layer packetization prior to the GA transport layer. The PES header carries various rate, timing, and data descriptive information, as set by the encoder. The PES packetization interval is application dependent. The resulting PES packets are of variable length with a maximum size of 2^{16} bytes, when the PES packet length field is set to its maximum value. This value is set to zero for the GA video stream, implying that the packet size is unconstrained and that the header information cannot be used to skip over the particular PES packet. This value of Note also that the PES packet format has been defined to also be of use as an input bit stream for Digital Storage Media (DSM) applications. Although the DSM format will not be used for GA broadcast application, some of the PES header fields related to the DSM functions are also described in this chapter. Note that the ability to handle input bit streams in the DSM format is not essential for a GA receiver, but may be useful for VCR applications.

The constraints on the length of the PES packets for GA video have to be settled upon. Among the decisions to be reached are whether PES packets should only start on GOP boundaries, whether all GOP boundaries should start a new PES packet, whether PES packets should be defined for each access unit (corresponding to a picture), etc.. Note that the format for carrying the PES packet within the GA transport is a subset of the general definition in MPEG. This choice was made to simplify the implementation of the GA receiver. Essentially, in the GA transport system, all data for a PES packet, including the header, are transmitted contiguously as the payload of transport packets. New PES packet data always starts a new transport packet, and PES packets that end in the middle of a transport packet are followed by stuffing bytes for the remaining length of the transport packet.

A PES packet consists of a PES_packet_start_code, PES header flags, PES packet header fields, and a payload (or data block), as shown in Fig. 6.1. It is created by the application encoder. The packet payload is a stream of contiguous bytes of a single elementary stream. For video and audio packets, the payload is a sequence of access units from the encoder. The access units correspond to the video pictures and audio frames.



NSDOCID: <XP_____2055175A_I_>

Figure 6.1. Structural Overview of a PES Packet

Each elementary stream is identified by a unique stream_id. The PES packets from each encoder carry the corresponding stream_id. PES packets carrying various types of elementary streams can be multiplexed to form a program or transport stream in accordance with Part 1 of the MPEG-2 standard. This chapter deals with the organization of PES packets for MPEG-2 compliant streams only. Specifically, the stream_id field can take on a number of values, indicating the type of data in the payload (See Table for valid values). This section does not define the PES packet structure for 'private data', i.e. stream_id must be 'Reserved', 'Padding', or 'MPEG Audio' or 'MPEG Video'. For 'private data', the PES packet structure is user defined.

The preliminary fields, the packet_start_code_prefix, stream_id, and PES_packet_length, are described in the table below.

Field	Description	GA Usage
packet_start_code_prefix	Indicates the start of a new packet. Together with the stream_id, it forms the packet_start_code. Takes on the value 0 x 00 00 01.	As defined
stream_id	Specifies the type and number of the stream, to which the packet belongs. 1011 1100 - Reserved Stream 1011 1101 - Private Stream 1 1011 1110 - Padding Stream 1011 1111 - Private Stream 2 110x xxxx - MPEG Audio Stream Number xxxxx 1110 xxxx - MPEG Video Stream Number xxxx 1111 xxxx - Reserved Data Stream Number xxxx	As defined
PES_packet_length	Specifies the number of bytes remaining in the packet after the this Field.	0 x 00 00 - Only allowed value for video. Details for audio yet to be determined

6.1. PES header Flags

A breakdown of the PES header flags is shown in Figure 6.2. These flags are a combination of indicators of the properties of the bit stream and indicators of the existence of additional fields in the PES header. The following table describes the flags present the header. The flags not supported by the GA system are set to '0' and form the basis of some of the "constraints" discussed earlier. (These entries are shaded in the table.) Note that the abbreviations in the tables are from the flag names in Fig. 6.2.

BNSDOCID: <XP_____2055175A_1_>

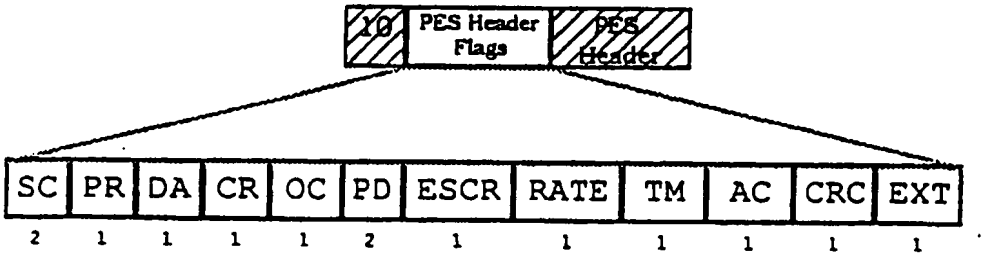


Figure 6.2. PES header flags in their relative positions (all sizes in bits)

NSDOCID: <XP_____2055175A_I_>

Flag	Description	GA Usage
PES_scrambling_control (SC)	Indicates the scrambling of the PES packet payload. 00 – Not Scrambled 01 – User Defined 10 – User Defined 11 – User Defined	Set to 00.
PES_priority (PR)	Indicates the priority of this packet with respect to other packets whose PR field is not set. 1 – Higher priority 0 – Same priority	GA does not care how this field is set. (It may be used by applications where necessary.)
data_alignment_indicator (DA)	Indicates the nature of alignment of the first start code occurring in the payload. The type of data in the payload is indicated by the data_stream_alignment_descriptor. 1 - Aligned 0 - No indication of alignment	Must be aligned for video. To be decided for Audio.
copyright (CR)	Indicates the copyright nature of the associated PES packet payload. 1 – Copyrighted 0 – Not copyrighted	The GA has yet to define its use of this field.
original_or_copy (OC)	Indicates whether the associated PES packet payload is the original program or a copy. 1 – Original 0 – Copy	The GA has yet to define its use of this field.
PTS_DTS_flags (PD)	This flag indicates whether the PTS or PTS and DTS are in the PES header. 00 – Neither PTS nor DTS is present in header. 1x – PTS field present 11 – PTS and DTS field present in header.	The PTS flag is set when video data alignment indicator is set. The DTS may be included to signal the decoder of any special decoding requirements. The PTS transmissions should be spaced less than 700 msecs apart.
ESCR_flag (ESCR)	Indicates whether the Elementary Stream Clock Reference field is present in the PES Header.	Set to '0'.
ES_rate_flag (RATE)	Indicates whether the Elementary Stream Rate field is present in the PES Header.	Set to '0'.
DSM_trick_mode_flag (TM)	Indicates the presence of an 8 bit field describing the mode of operation of the DSM (Digital Storage Media). 1 – Field present 0 – Field not present	Set to '0' for the broadcast transmission. May be used for trick modes when using bit streams specifically generated for VCR type operations.
additional_copy_info_flag (AC)	Indicates the presence of the additional_copy_info field. 1 – Field Present 0 – Field not present	The GA has yet to define its use of this field.
PES_CRC_flag (CRC)	Indicates whether a CRC field is present in the PES packet.	Set to '0'.
PES_extension_flag (EXT)	This flag is set as necessary to indicate that extension flags are set in the PES header. Its use includes support of private data. 1 – Field present 0 – Field not present	As defined.

9NSDOCID: <XP_____2055175A_1_>

6.2. The PES header

The PES header immediately follows the field PES_header_length, which indicates the header size in bytes. The size of the header includes all the header fields, any extension fields, and stuffing_bytes. The flags described in the previous section indicate the organization of the PES header, i.e. which fields it does and does not contain. In essence, all the fields of the PES header are optional. Certain applications require particular fields to be set appropriately. For example, GA transport of video PES packets requires that the data_alignment_indicator be set. The trick mode flag is not set in this case. For DSM retrieval of video, the opposite is true. It is the application encoder's function to set the appropriate flags, and encode the corresponding fields. The fields are further described in the following sections. The association between the flags and the corresponding fields is obvious.

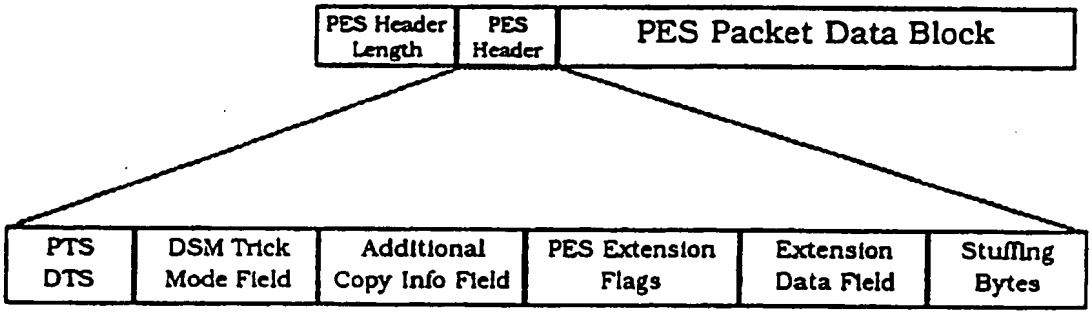


Figure 6.3. Organization of PES header.

The PES header Fields are organized according to Fig. 6.3 for the GA PES packets for video elementary streams. Most fields require marker bits to be inserted, as described later, in order to avoid the occurrence of long strings of 0's which could resemble a start code.

PTS and DTS

The presentation_time_stamp (PTS) informs the decoder of the intended time of presentation of a presentation unit, and the decoding_time_stamp (PTS) is the intended time of decoding of an access unit. An access unit is an encoded presentation unit. When it is encoded, the PTS refers to the presentation unit corresponding to the first access unit occurring in the packet. If an access unit does not occur in a PES packet, it shall not contain a PTS. Here, a video access unit is said to occur if the first byte of the picture start code is present in the PES packet payload, and an audio access unit occurs if the first byte of the synchronization word of an audio frame is present. Under

INSDOCID: <XP_____2055175A_I_>

normal conditions, the DTS may be derived from the PTS. So, it is not required to encode the DTS. Consequently, encoding the DTS may indicate special decoding requirements to the decoder. Under no circumstance does the DTS occur by itself; it must occur along with the PTS although the converse is not true. The PTS field is organized as shown, if it present without the DTS.

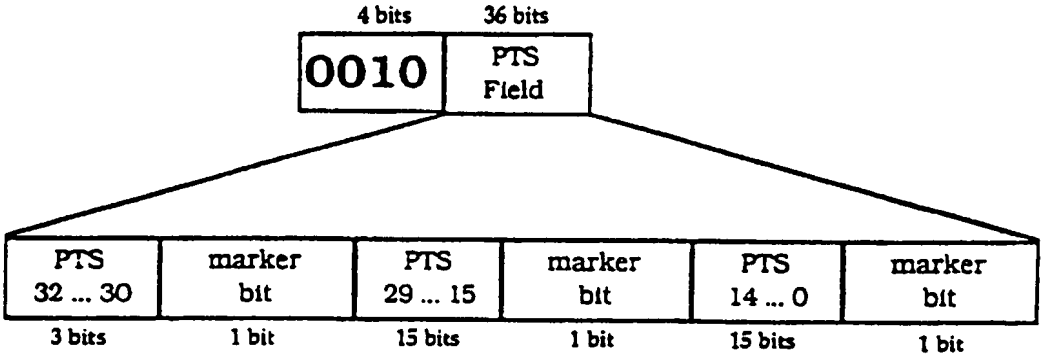


Figure 6.4. Organization of the PTS field when only the PTS is encoded.

If both the PTS and DTS are sent, the following organization is required.

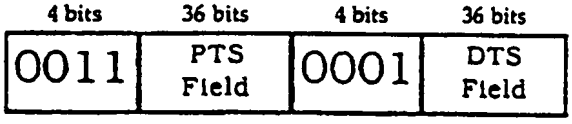


Figure 6.5. Organization of the PTS and DTS field when both PTS and DTS are encoded.

Here, the DTS field is defined in the same manner as the PTS field.

DSM Trick Mode Field[5]

The DSM trick mode field is an eight bit field, indicating the nature of the information encoded in the PES packet. The first three bits of this field form the identifier trick_mode_control, indicating the nature of the DSM mode. There are four modes to the DSM, as summarized in the table below.

[5] It is emphasized once again that the ability to deal with trick mode bit streams is not a basic requirement for a GA receiver and this description is here only for completeness. This field is not present in the broadcast GA bit stream. Since the objective in this document is not to describe in detail the DSM mode of operation, all that is presented is the DSM syntax with some description of the respective fields. For more detailed information the reader is directed to the MPEG-2 Systems document.

INSDOCID: <XP_____2055175A_I_>

Value	Description
'000'	Fast Forward
'001'	Slow Motion
'010'	Freeze Frame
'011'	Fast Reverse
'1xx'	Reserved

DSM Trick Modes

Depending on the value of trick_mode_control, a combination of four identifiers are encoded as described in the table below.

Identifier	Description
field_id	This identifier is valid for interlaced pictures only. It is a 2 bit field which identifies how the current frame is to be displayed. '00' – Display field 1 only '01' – Display field 2 only '10' – Display complete frame '11' – Reserved
frequency_truncation	This 2 bit field indicates the selection of coefficients from the DSM. '00' - Only DC coefficients are sent '01' - The first three coefficients in scan order on average '10' - The first six coefficients in scan order on average This field is for informational purposes only. i.e. the DSM may at times send more than the specified number of coefficients and at other times less. However, that information is not normative.
intra_slice_refresh	This 1 bit field indicates that each picture is composed of intra slices with possible gaps between them. The decoder should replace the missing slices by repeating the colocated sites from the previous decoded picture.
field_rep_cntrl	This field indicates how many times the decoder should repeat field #1 as both top and bottom fields alternatively. After field #1 has been displayed, the decoder should repeat field #2 the same number of times. This identifier being set to 0 is equivalent to a freeze frame with field_id being set to '10'.

Fast Forward and Fast Reverse Modes: The format in this case is shown in Fig. 6.6. The decoder is told how many coefficients are encoded, how the fields are to be displayed, and how to replace any missing slices in the access units.

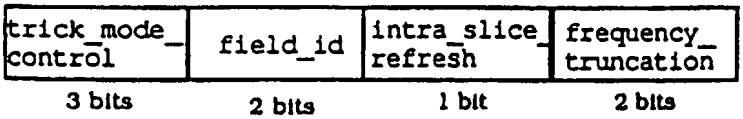


Fig. 6.6. Trick Mode Field in Fast Forward and Fast Reverse Modes

Slow Motion Mode: In this mode, the DSM informs the decoder how many times a particular field is to be repeated. The identifier field_rep_cntrl is encoded as shown.

NSDOCID: <XP_____2055175A_I_>

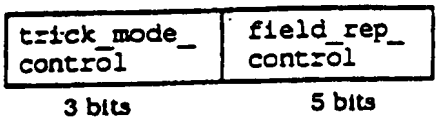


Fig. 6.7. Trick Mode Field in Slow Motion Mode

Freeze Frame Mode: Only the identifier is encoded in this mode. It indicates to the decoder how to display the frozen picture. The field is organized as shown in Figure 6.8.

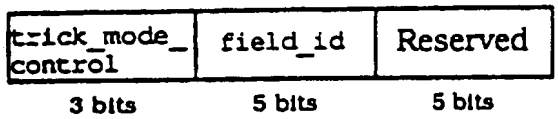


Fig. 6.8. Organization of Trick Mode Field for Freeze Frame Mode.

Additional Copy Info

This is a one byte field with a marker bit up front and 7 bits of information. The use of these seven bits is yet to be determined.

PES Extension Flags

The header could contain additional flags if the EXT flag (shown in Fig. 6.2) is set. These flags are transmitted in a one byte data field as shown in Fig. 6.9.

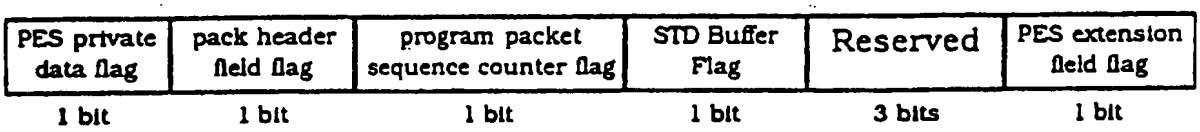


Figure 6.9. Organization of the PES Extension Flags Field.

The flags indicate whether further extensions to the PES header exist. The table below describes the nature of this additional data. As with the flags defined previously, the flag is set to '1' if the header field is present.

Field	General Description
PES_private_data_flag	Indicates whether the PES packet contains private data.
pack_header_field_flag	Indicates whether an MPEG-1 systems pack header or an MPEG-2 program stream pack header is present in the PES Header.
program_packet_sequence_counter_flag	Indicates the presence of a PES packet counter, which allows the decoder to retrieve the packets in the correct order, from the PES/program multiplex.
STD_buffer_flag	Indicates whether the STD_buffer_scale and the STD_buffer_size flags are encoded in the PES header.
PES_extension_field_flag	Indicates the presence of additional data in PES header.

BNSDOCID: <XP_____2055175A__1_>

For the GA system, as indicated in the constraint document submitted to MPEG, the flags that are shaded in the table are always set to '0'. It is likely that the EXT flag will set to '0' in the header flags field unless there is information to be transmitted in the PES_extension_field. This field is mean for functions that may have been missed in the initial design specification.

PES Extension Field

This field is shown in Fig. 6.10. The length of the PES_extension_field data is given by PES_extension_field_length.

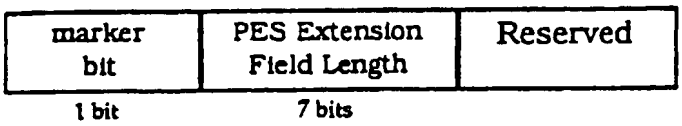


Figure 6.10. Organization of the Extension Field

INSDOCID: <XP_____2055175A_I_>

7. Conditional Access

The transport protocol implements functions useful for supporting conditional access. The functionality that is available is flexible and complete in the sense of supporting all transmission aspects of applicable key encryption and descrambling approaches that may be used. Conditional access is also flexible in the sense that it can be exercised on a elementary stream by stream basis, including the ability to selectively scramble bit streams in a program if desired.

A conditional access system operates on the principle of randomizing the transmitted data so that unauthorized decoders cannot decode the signal. Authorized decoders are delivered a "key" which initializes the circuit which inverts the bit randomization. In subsequent discussion, we use the term scrambling to mean the pseudo-random inversion of data bits based on a "key" which is valid for a short time. We use the term encryption to mean the process of transforming the "key" into an encrypted key by a means which protects the key from unauthorized users. From a cryptographic point of the view, this transformation of the key is the only part of the system which protects the data from a highly motivated pirate. The scrambling portion of the process alone, in the absence of key encryption, can be defeated. Conditional Access (CA) is a blanket term for the system which implements the key encryption and distribution. The primary requirements which a scrambling and CA subsystem must meet for digital TV delivery are:

- Protection of programmer's revenues
  - robust against piracy.
- Private encryption system for each program provider.
- Standard consumer instruments
  - no secrets in consumer equipment
- Mobility of consumer equipment
- Consumer equipment should be cost effective

INSDOCID: <XP_____2055175A_I_>

7.1. General Description

There are two features of the GA Transport system which support conditional access. The first feature is the two bit transport_scrambling_control field which signals the decoder whether the transport packet was scrambled or not. In the case that it was scrambled, the field identifies which scrambling key was used. As will be shown shortly, the use of two bits in the transport_scrambling_control to define the descrambling process is a necessary and sufficient bound for the key distribution function. The second feature is the ability to insert "private" data at several places in the GA Transport stream. These include entirely private streams and private fields in the adaptation header of the transport bit stream being scrambled. These private fields can be used to transmit the encrypted scrambling key to the decoding device.

The key distribution and usage process is clarified in Fig. 7.1. Basically, when the bit stream is scrambled, one descrambling key needs to be in use while the other is being received and decrypted. Two keys are transmitted at any time, with the keys being linked to a transport_scrambling_control value as shown in the figure. The transmission of a key should begin well before it is going to be used, to allow time to decrypt it. Note that this function does not bound the total number of keys that may be used during an entire transmission session.

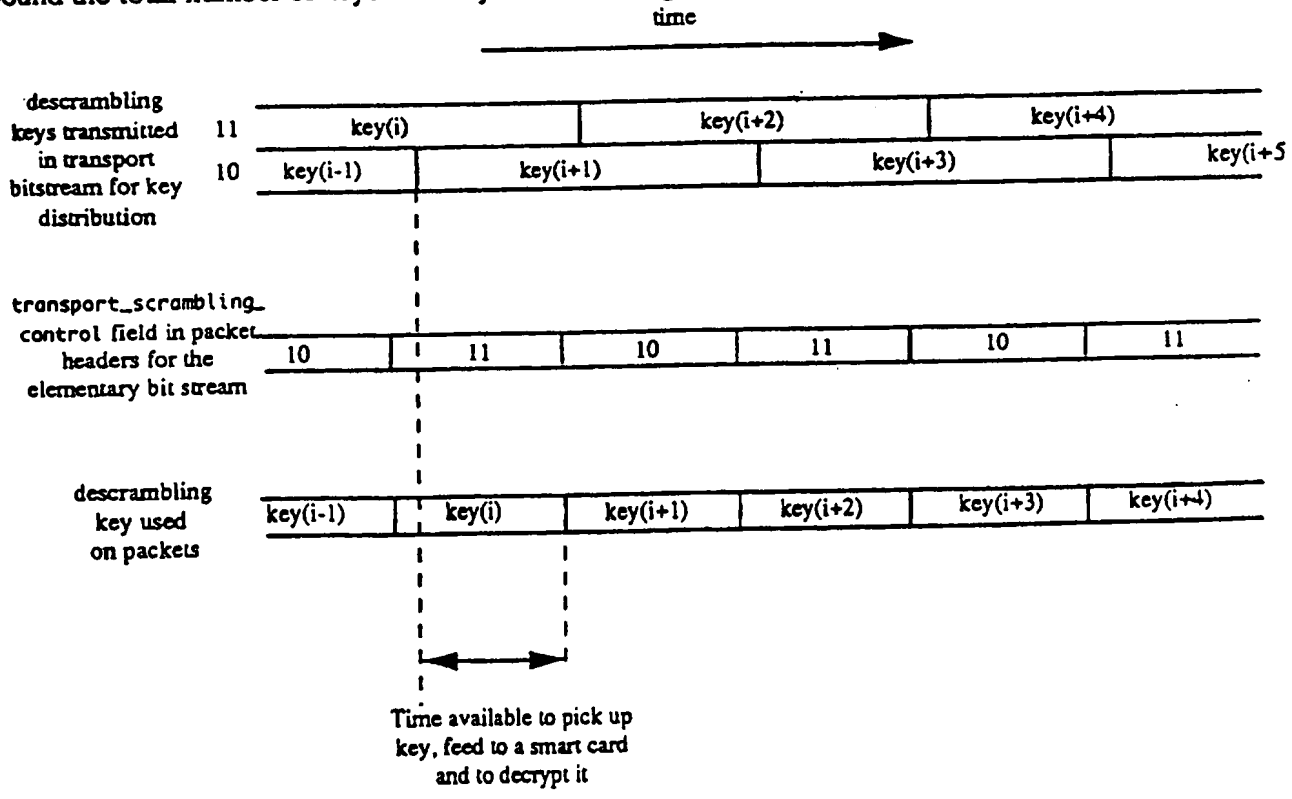


Fig. 7.1. Illustration of key distribution and usage process.

BNSDOCID: <XP_____2055175A__I_>

As stated previously, the amount of data to be scrambled in a packet is variable depending on the length of the adaptation header. It should be noted that some padding of the adaptation field might be necessary for certain block mode algorithms.

7.2. Example of Conditional Access Implementation

In this section, we go through a simple example of a conditional access implementation. Consider the receiver architecture shown in Figure 7.2. The high speed manipulations required to implement the descrambling are embedded in the transport demultiplexer, where they are shown as a DES block. Note that other scrambling schemes, such as stream ciphers based on a Pseudo Random Binary Sequence (PRBS), could be employed. The PRBS uses a shift register implementation, where it the initial register value is reset periodically for error robustness. The data security is achieved by the "key" which properly configures the descrambler. This element is delivered to the decoder through an ancillary data service, and is encrypted by the conditional access administrator. In the equipment at the customer's premises, the key is decrypted within the outboard Smart-Card. The Smart-Card interface will conform to ISO standard ISO-7816, which permits a variety of implementations and conditional access solutions.

The Smart-Card maintains a short list of two key's, commonly denoted as the "odd" key and the "even" key. The proper key to be used to descramble is signaled in the transport prefix in the transport_scrambling_control field. The transport_scrambling_control takes on one of the following 4 states:

transport_scrambling_control	*Description*
00	Not Scrambled
01	Reserved
10	"even" key
11	"odd" key

The scrambling in this example is a block cipher called the "Electronic Code Book" mode of the Digital Encryption Standard (DES). For DES, the key is a 56 bit binary sequence. The television electronics are "standard", while the Smart-Card implementation is proprietary to the service provider.

BNSDOCID: <XP_____2055175A_1_>

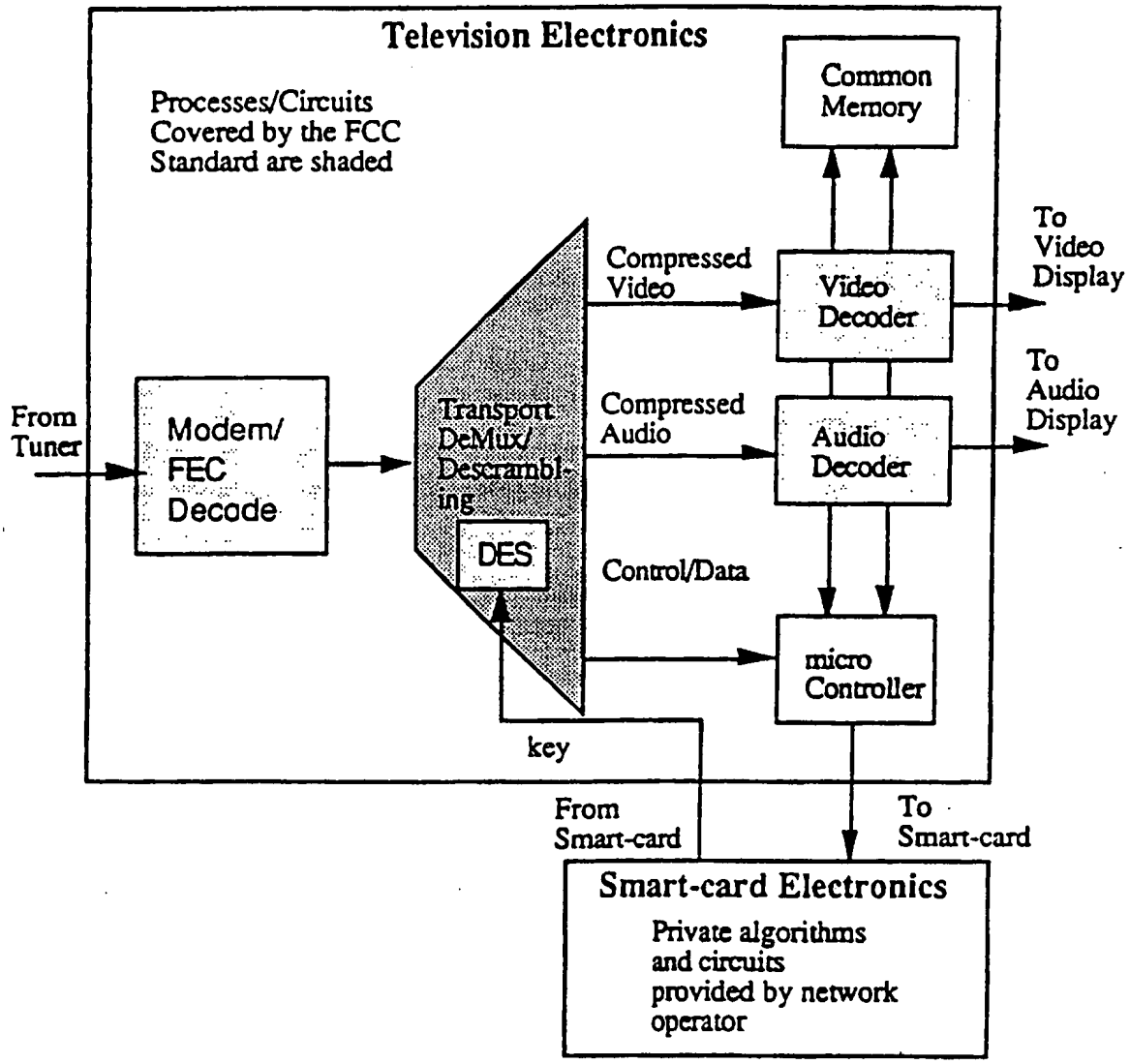


Fig. 7.2. Decoder with Example Conditional Access Implementation

There is significant flexibility for multihop encryption and nesting of encryption systems to ensure data security at every point in the transmission chain. Fig. 7.3 illustrates nested encryption systems, where the service provider provides authorization, keys and the scrambled data to the end user through Encryption System A. During transit, System B is employed by the carrier to protect the data while in the communications network. In fact, there are two implementation choices even for this segment of the delivery system. The provider can use both a second layer of scrambling in conjunction with a different authorization and key distribution. (This system need not comply with the transmission "Standard's" method.) Alternatively, System B could simply encrypt or scramble the encrypted keys distributed by System A, without the requirement of scrambling the actual

INSDOCID: <XP_____2055175A_I_>

service data. The main appeal of this method is that it is a low bandwidth/complexity solution, while its drawback is that it requires some knowledge of the System A key distribution method.

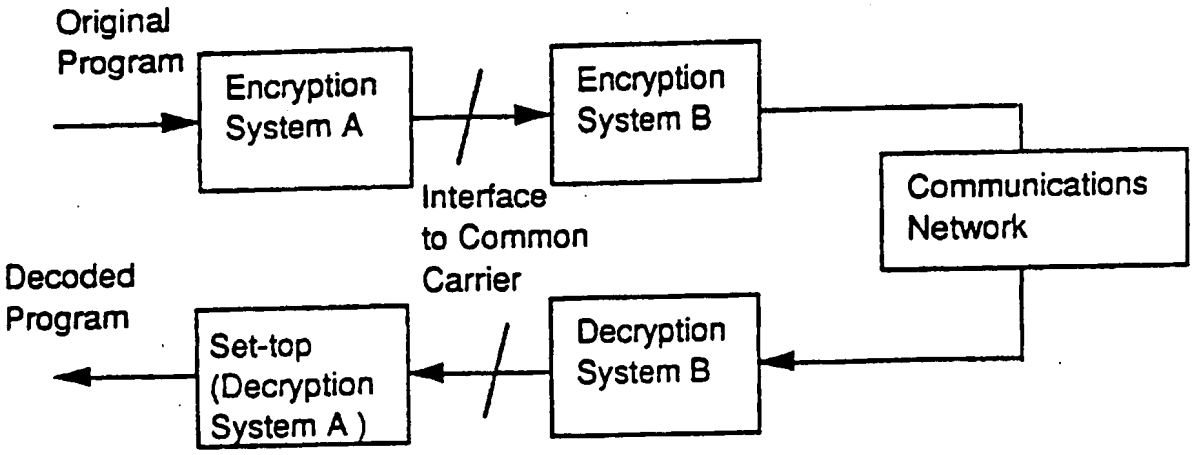


Figure 7.3. Nested encryption systems.

In Fig. 7.4, the two Encryption Systems are connected in series. System B is again used to protect the integrity of the data while in the communications network. System A is used by the local affiliate or cable company to authorize reception of the service within its own service area.

In fact, the two topologies discussed above can be combined, where either System A or System B could be a Nested or Series configuration. The number of nesting/series combinations can be arbitrarily large.

BNSDOCID: <XP_____2055175A__I_>

Transport System

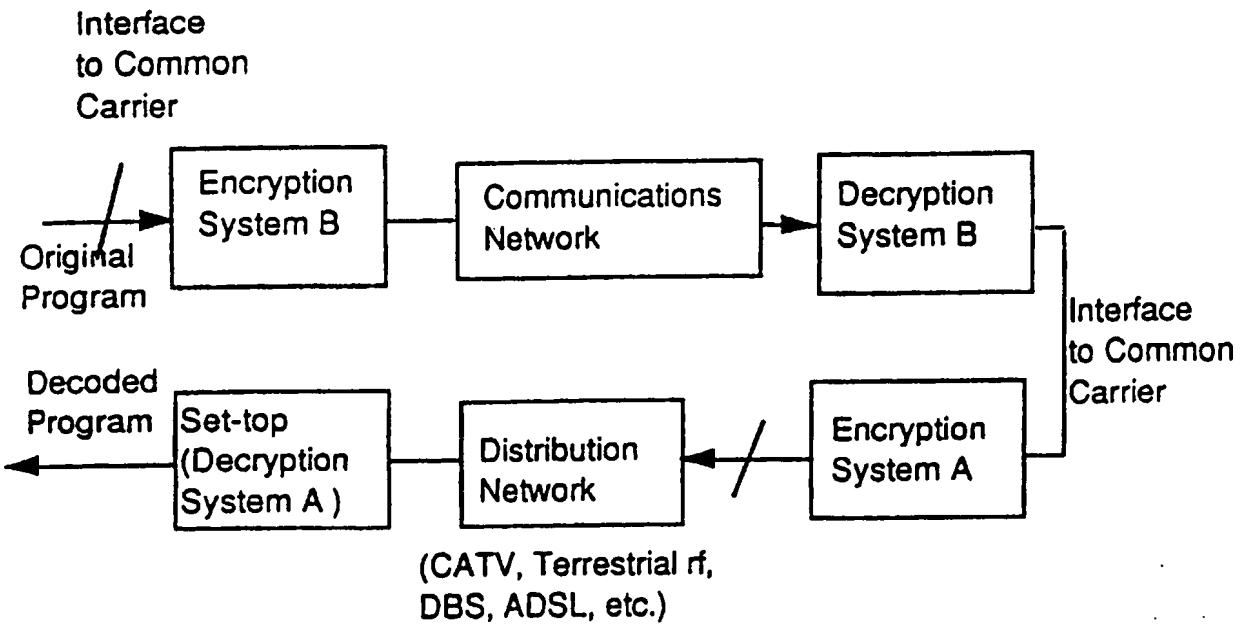


Figure 7.4. Series encryption systems.

INSDOCID: <XP_____2055175A_I_>

8. Local Program Insertion

The Grand Alliance Transport supports insertion of programs and commercials, by use of flags and features dedicated to this purpose in the transport packets Adaptation Header. The use of these syntax elements will need to be within some imposed constraints to ensure proper operation of the video decoders. Furthermore, there will be some constraints on some of current common practices, imposed not by the GA transport, but rather by virtue of the compressed digital data format.

The functionality of program insertion and switching of channels at a broadcast head-end are quite similar, the difference being in the time constants involved in the splicing process, and also in the fact that in the program insertion the bit stream is switched back to the old program after insertion is complete, while in the channel switching case one most likely switches over to yet another program at the end of the splice. There are other detailed issues related to the hardware implementation that may differ for these two cases, including input source devices and buffering requirements. For example, if program insertion is to take place on a bit stream obtained directly from a network feed, and if the network feed does not include place-holders for program insertion, the input program transport stream will need to be buffered up for the duration of the program insertion. If the program is obtained from a local device, e.g., a video server or a tape machine, it may be possible to pause the input process for the duration of the program insertion. Neither of these is an issue for channel switching.

8.1. Systems level view

There are two layers of processing functionality to address when doing program insertion. The lower layer functionality is related to splicing of transport bit streams for the individual elements of the program. The higher level functionality is related to coordination of this process between the different elementary bit streams which make up the program transport stream. Fig. 8.1 illustrates the correct approach to implement program insertion.

INSDOCID: <XP_____2055175A_I_>

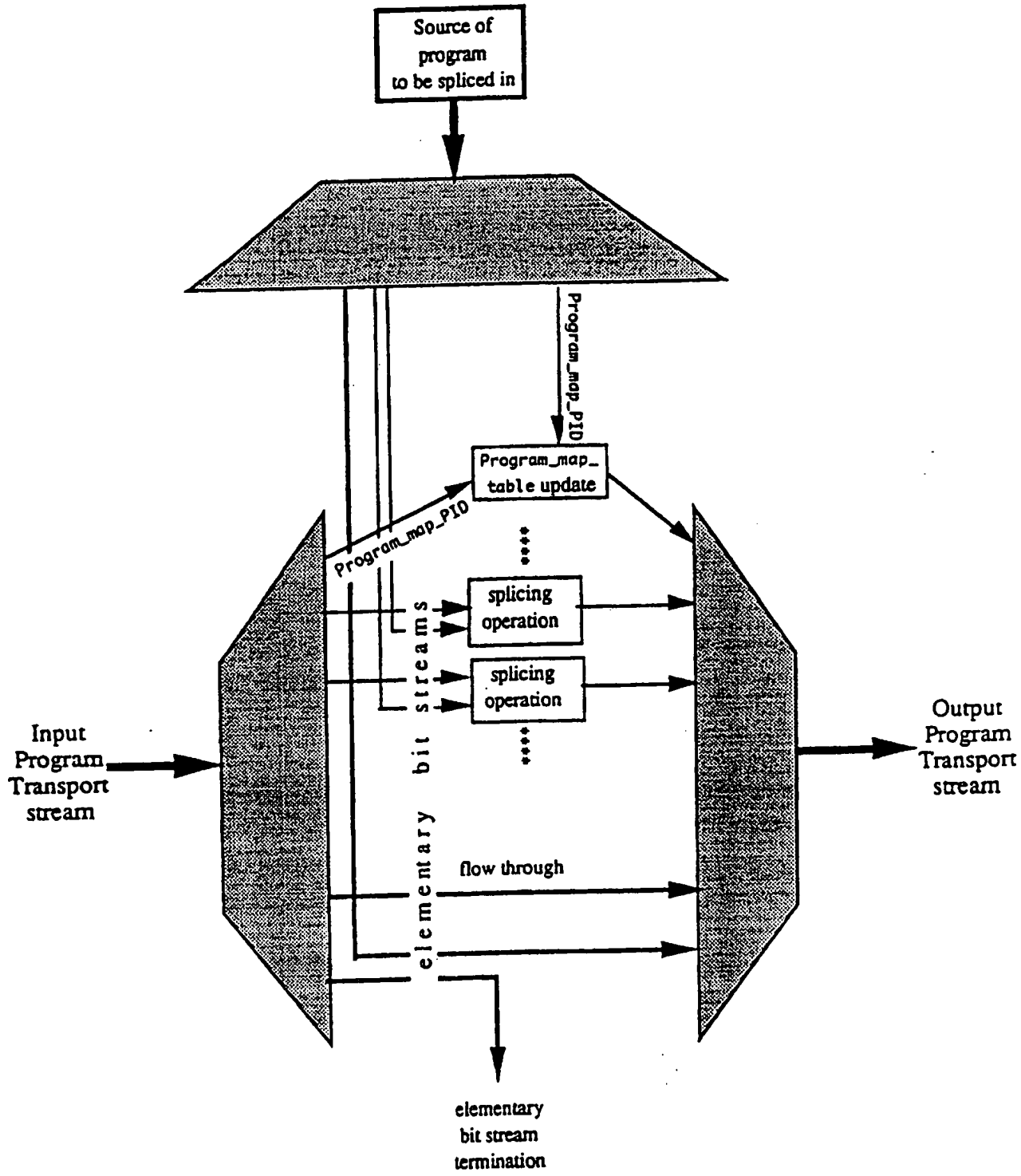


Figure 8.1 Example Program Insertion Architecture

The first step for program insertion to take place at a broadcast head-end is to extract (by demultiplexing) the packets, identified by the PIDs, of the individual elementary bit streams that

NSDOCID: <XP_____2055175A_I_>

make up the program, including the bit stream carrying the program_map_table. Once these packets have been extracted, as illustrated in Fig. 8.1, program insertion can take place on an individual PID basis. If applicable, some packets may be passed through without modification. There is also the flexibility to add and drop elementary bit streams. The splicing process for each PID is described in the next section.

When program insertion takes place, the program_map_table needs to be modified to reflect the properties of the program transport stream that is being spliced in. As described in the section on its syntax, the definition of the program_map_table allows the signaling of a change in the contents of a program transport stream ahead of time. The changes in the program definition could involve a change in the number of elementary bit streams that make up the program, either by addition or removal of bit streams, change in the PIDs used for the elementary bit streams, etc...

The bit-rate of the program after splicing should have a known relationship to the bit-rate of the program before splicing, in most scenarios. Unless dynamic bit-rate allocation is possible for a program at the system multiplexer (based on instantaneous bandwidth requirements), an increase in bit-rate after splicing can cause buffer overflow. A decrease in bit rate may be handled by transmitting null packets (packets with no information) or by allocating the extra bandwidth to other programs on a dynamic basis. These capabilities depend on the implementation of the system level multiplexing function, a function that is not a part of the GA Transport specification.

Bit rate constraints may also be imposed on individual elementary bit streams for the program that is inserted, e.g., for compressed video, input and output bit rates need to be the same. In a perfect program insertion set up, the splice points for the different elementary streams in a program should be coordinated to correspond to the same instant in time in the overall program (which may not correspond to the same instant in time for each elementary bit stream) to permit seamless transition. Additional constraints on selecting the splicing points exist for particular applications such as video (i.e., VBV_delay value).

8.2. Basics of elementary bit stream insertion

The interface for elementary bit stream insertion is at the transport layer of the protocol. This means that bit stream insertion always takes place in units of transport packets. The primary features enabling local elementary bit stream insertion are the discontinuity_indicator field and the splice_countdown fields in the transport header. The discontinuity_indicator signals the decoder that the PCR is changing to a new time base. This simply informs the decoder that the change in the bit stream is not due to an error in the channel, but rather is intended by the program provider. The

BNSDOCID: <XP_____2055175A_1_>

implication for the decoder is that it should continue normal decoding, and it is the encoders responsibility to make sure that the bit stream has been constructed in a compliant manner (that is that decoders don't crash due to overflow or underflow.)

The splice_countdown field in the adaptation header is used to signal a head-end or intermediate digital switch that a subsequent packet is the point for switching in a new bit stream. The count-down is a positive number which decrements on each subsequent packet of that service. The value of "0" is the last packet in the original sequence, and the value "-1" is resident in the packet which should initiate the switch over. The count will continue to decrement for channel error resilience. The behavior is shown in Figure 8.2.

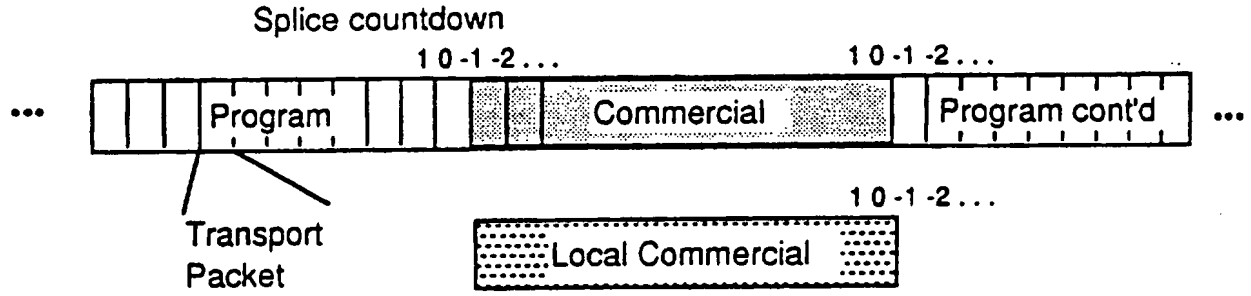


Figure 8.2. Local program insertion keyed on Splice countdown.

The affiliate or headend equipment sets itself up for the switch based on the descending countdown. At the "-1" point, the local commercial is inserted while the network commercial continues. At the second trailing "-1", the affiliate returns to the network feed. This technique can be used for either local programs or local commercials. It does depend on the video encoder constraining its bit generation at the splice points so that the decoder buffer does not overflow.

The GA transport encoder places some constraints on the encoding which are more stringent than the MPEG-2 requirements. The added constraint is that the PES header is followed immediately by a video access unit. This will speed acquisition. The first packet in an insertion will contain the PCR value, with the PCR discontinuity bit set to "1" to inform the decoder that a splice has occurred. The first payload in the stream will begin with a PES Header, which will have a PTS resident, so that the decoder can determine the display time immediately. Because the PES header also has the data_alignment_indicator set, the first data following the header will be the start of the video sequence layer. Consequently, the decoder has all the information available to begin

3NSDOCID: <XP_____2055175A__I_>

decoding immediately after receiving the beginning of the spliced commercial. (In general, an MPEG-2 stream does not have these constraints imposed, and hence does not have guaranteed performance at the splice points.)

8.3. Restrictions

Compressed digital technology does impose some restrictions on affiliate operations which may differ from present practice. Although all present practices can be replicated by completely decoding and recoding the video, there is a desire to implement as much as possible in the compressed video domain. The following comments are made with respect to processing the compressed video.

Local commercials and network commercials will need to be strictly controlled to be the same number of packets, and the same number of frames of video. This is contrary to the present practice where local inserts may differ from the planned network inserts by several seconds.

A second restriction is that affiliate pix-in-pix and affiliate text overlays can only be accomplished by decoding and recoding.

8.4. Imperfect program insertion

It may not always be possible for the program insertion process to meet the precise requirements of a seamless splice. This could be due to several reasons including the presence of infrequent splice points in the incoming bit stream or non-availability of the hardware required for precise splicing at the network affiliate. There are two scenarios for imperfect splicing. In the first scenario the network affiliate attempts to splice in the entire program as a whole, without attempting to align each of the component elementary bit streams. In this case, the exact splicing can take place for only one of the elementary streams. Since video is the most important component of the program, perfect alignment will be obtained for video. In this case the output of the other elementary bit streams will not be presented at the output until synchronization of these bit streams is achieved. As an example, audio should be muted until its elementary bit stream is synchronized.

In the second and most uncoordinated splicing approach, the splicing takes place without any attempt at coordination with the input bit stream. In this case the video presentation process is affected around the splicing point. If the splicing takes place when the VBV level in the existing bit stream is less than it should be for perfect splice, there will be a period of time for which data is lost for the existing bit stream. In this case the decoder should freeze the last displayed frame. In the other case where VBV is fuller than expected, the decoder video data buffer may eventually

NSDOCID: <XP_____2055175A_I_>

overflow during the time period of the spliced in bit stream. The decoder will then have to initiate a resynchronization procedure in the middle of the program, freezing the display to the last decoded picture while this process is taking place. Note that when the process of splicing in a bit stream does not take place correctly, there will also be a disruption in service at the splice back to the original bit stream. It is the recommendation of the GA that this type of splicing be strictly prohibited, since it leads to a very noticeable interruption of service.

It is important to note that the process of facilitating frequent opportunities for splicing in a program bit stream is not within the control of the transport layer of the system. The transport only provides the mechanism of implementing the splice itself. Hence decisions on determining the possible frequency of commercial insertion should also involve the people involved in the design of the source coding algorithms for applications like video and audio.

3NSDOCID: <XP_____2055175A__I_>

9. Compatibility with other Transport Systems

The GA transport system is compatible with two of the most important alternative transport systems, namely the MPEG-2 transport stream definition, and also the ATM definition being finalized for Broadband ISDN. Furthermore, since several of the CATV (e.g., Digicipher II) and DBS systems being designed are considering use of the MPEG-2 Transport layer syntax, the degree of interoperability with such deployed systems should be quite high (possibly requiring a translation if the CATV or DBS system deploys a slightly incompatible MPEG-2 variant).

9.1. Interoperability with MPEG-2

In the development of the GA transport specification, the intent has never been to limit the design by the scope of the MPEG-2 systems definition. The GA system is interoperable with MPEG-2 decoders since the GA Transport is currently a constrained subset of the MPEG-2 Transport syntax. The constraints are imposed for reasons of increased performance of channel acquisition, bandwidth efficiency and decoder complexity. If, in the course of future work, the MPEG-2 standard is unable to efficiently meet the requirements of the GA system, a deviation from MPEG would be in order.

The ATV system requires definition of bit streams and services beyond the compressed video and audio services. A means of identifying such bit streams is necessary in the ATV system, but is not part of the MPEG-2 definition. There is a method of encoding such a registration descriptor when a an authority to administrate registration is identified. This identification is implemented by the registration_descriptor in the PSI stream.

9.2. Interoperability with ATM

The GA transport packet size is selected to ease transferring these packets in a link layer that supports Asynchronous Transfer Mode (ATM) transmission. There are several methods for mapping the Transport packet into the ATM format. Three techniques are presented, although the industry may converge to a different solution than those presented here.

9.2.1. ATM Cell and Transport Packet Structures

Figure 9.1 shows the format of an ATM cell. The cell consists of two parts: a five byte header and a forty-eight byte information field. The header, primarily significant for networking purposes, consists of the following fields:

BNSDOCID: <XP_____2055175A_I_>

GFC	a four bit Generic Flow Control field used to control the flow of traffic across the User Network Interface (UNI). Exact mechanisms for flow control are under investigation.
VPI	an eight bit network Virtual Path Identifier.
VCI	a sixteen bit network Virtual Circuit Identifier.
PT	a three bit Payload Type (i.e., user information type ID).
CLP	a one bit Cell Loss Priority flag (eligibility of the cell for discard by the network under congested conditions).
HEC	an eight bit Header Error Control field for ATM header error correction
AAL	ATM Adaptation Layer bytes (user specific header).

The ATM User Data Field consists of forty-eight bytes, where up to four of these bytes can be allocated to an Adaptation Layer.

Figure 9.2 shows the format of the Grand Alliance transport packet. A one hundred eighty-four byte packet data field (possibly including an optional and conditional adaptation field) is preceded by a four byte prefix.

9.2.2. Null AAL Byte ATM Cell Formation

The simplest method to form ATM cells from the Transport layer is the null AAL byte structure shown in Figure 9.3. The Transport packet is partitioned into forty-eight byte payloads, applied directly to the information fields of the ATM cell. The five byte ATM header is appended. Since the Transport packet length is not an integer multiple of the ATM cell payload, there will be only occasional alignment of the Transport header with the start of the ATM cell information field.

9.2.3. Single AAL Byte ATM Cell Formation

Alignment of the Transport packet and ATM cell is accommodated by parsing the Transport packet into forty-seven byte segments, shown in Figure 9.4. Four such segments will exactly encompass a Transport packet. A one byte AAL is appended, along with the five byte ATM header to fulfill the fifty-three byte ATM cell requirement. The AAL byte can carry useful information concerning the transport data within the ATM cell. It can be viewed as an adaptation field for the contained data, conveying the original position of the ATM payload within the Transport packet, for example, as well as other information. For example, ATM standards presently provide for five different AALs, such as AAL Type 1 for accommodating connection oriented constant bit rate services, and AAL Type 2 for handling connection oriented variable bit rate data services.

9.2.4. Dual AAL Byte ATM Cell Formation

An alternative solution to cell/packet alignment is shown in Figure 9.5. The transport header is discarded, and the remaining one hundred eighty-four byte payload is segmented into 46 byte increments. To these are added two AAL bytes and the five byte ATM header for each ATM packet. The idea here is that there may be a duplication in functionality in the ATM header and the link level transport header fields. A particular header field to consider for duplication of functionality is the PID. If the PID can be associated with a specific VPI and VCI used in the ATM headers of the packets carrying the data payload, and this PID mapping information can be sent to the destination terminal when the virtual path/circuit is set up (using the ATM signaling channel), it does not have to be transmitted for every GA packet. The PID can then be reconstructed at the destination (using the information transmitted at call setup), and can then be appended to the one hundred eighty-four byte payload (reconstructed from four ATM packets) to obtain the complete GA packets. Transport header information that cannot be reconstructed (e.g., adaptation field control) should be carried as a part of the two AAL bytes for each ATM cell, along with other additional information. Note that the above is only a suggested approach and does not represent a complete design.

BNSDOCID: <XP_____2055175A__I_>

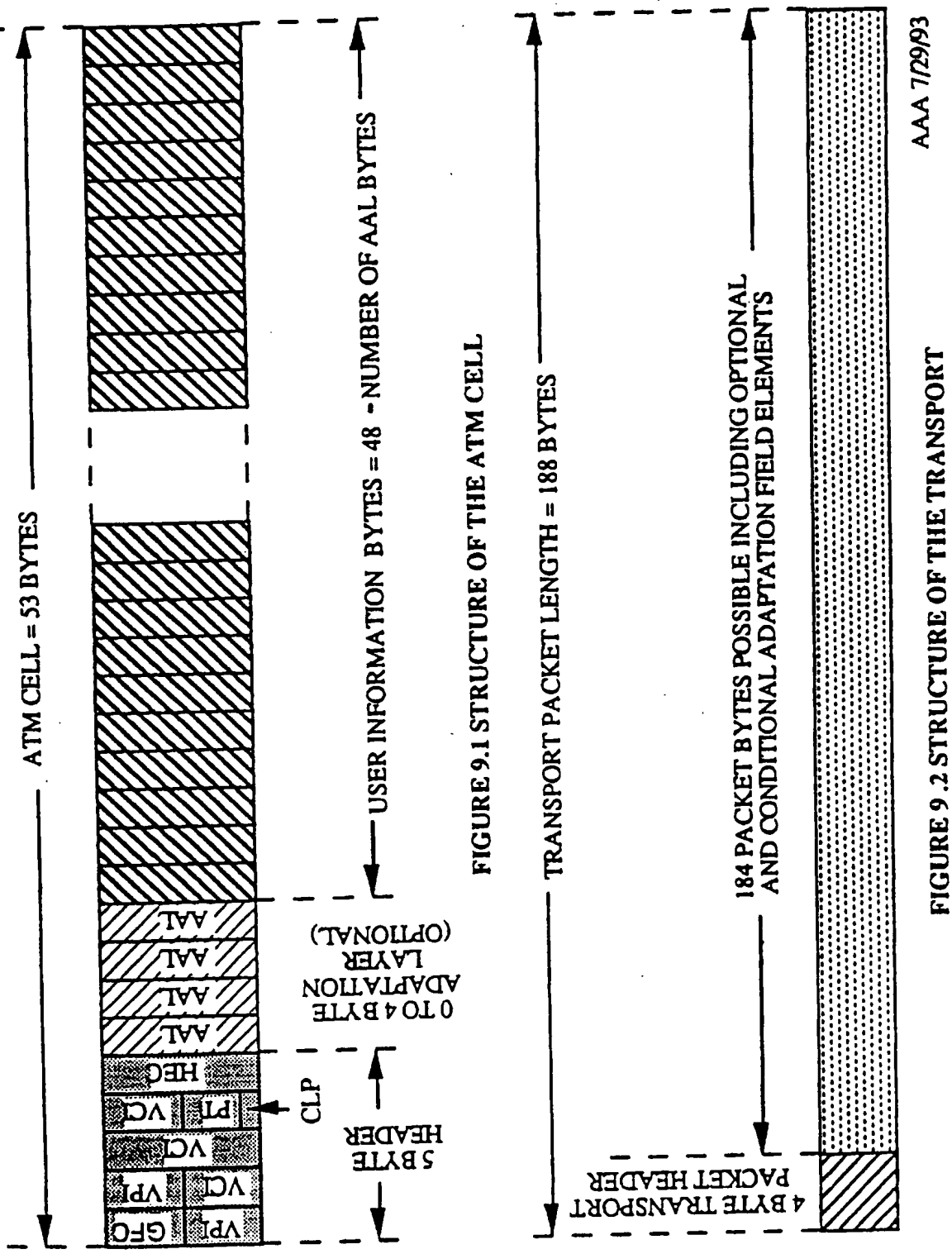


FIGURE 9.1 STRUCTURE OF THE ATM CELL

FIGURE 9 .2 STRUCTURE OF THE TRANSPORT PACKET

INSDOCID: <XP_____2055175A__I_>

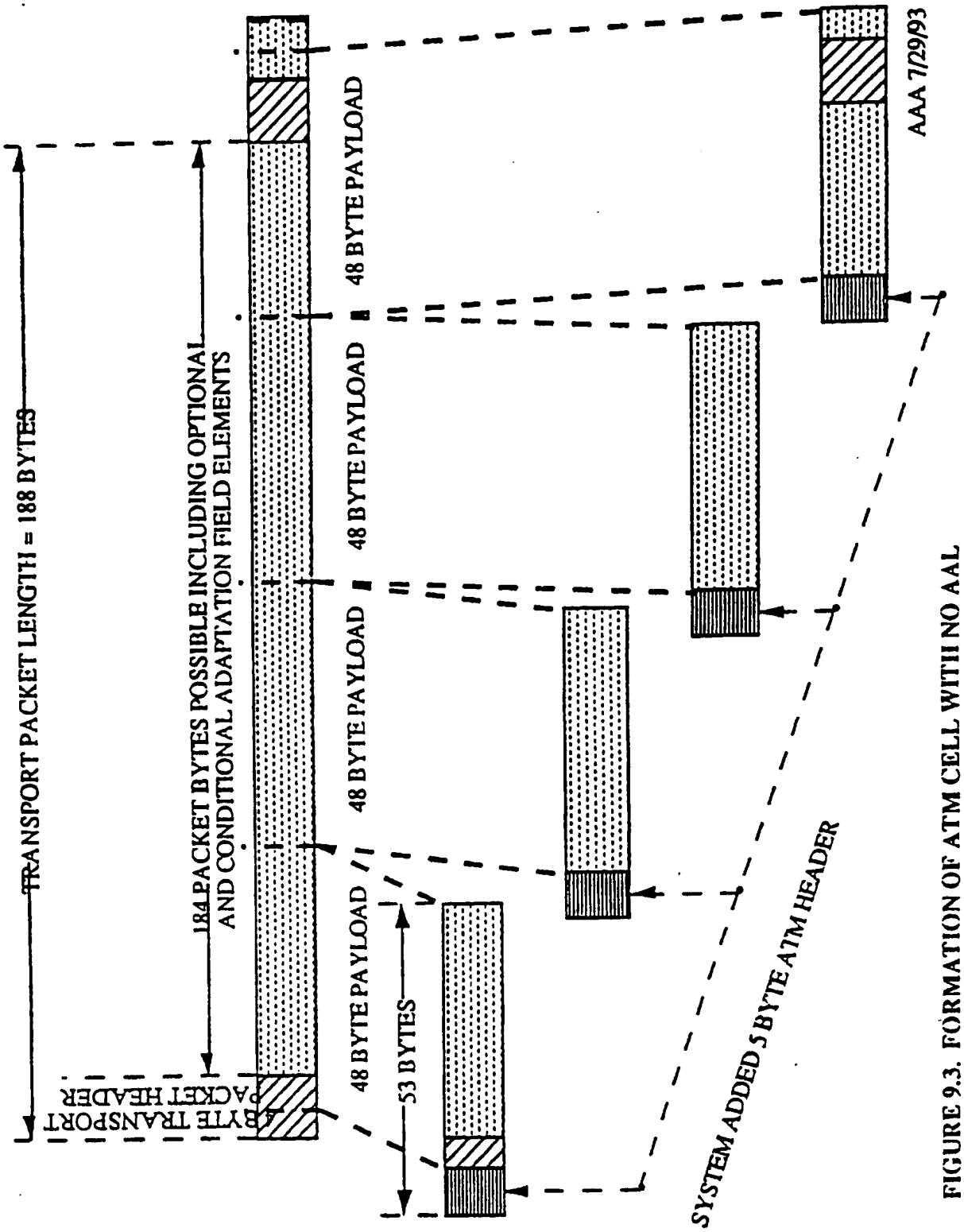


FIGURE 9.3. FORMATION OF ATM CELL WITH NO AAL BYTES

BNSDOCID: <XP_____2055175A__I_>

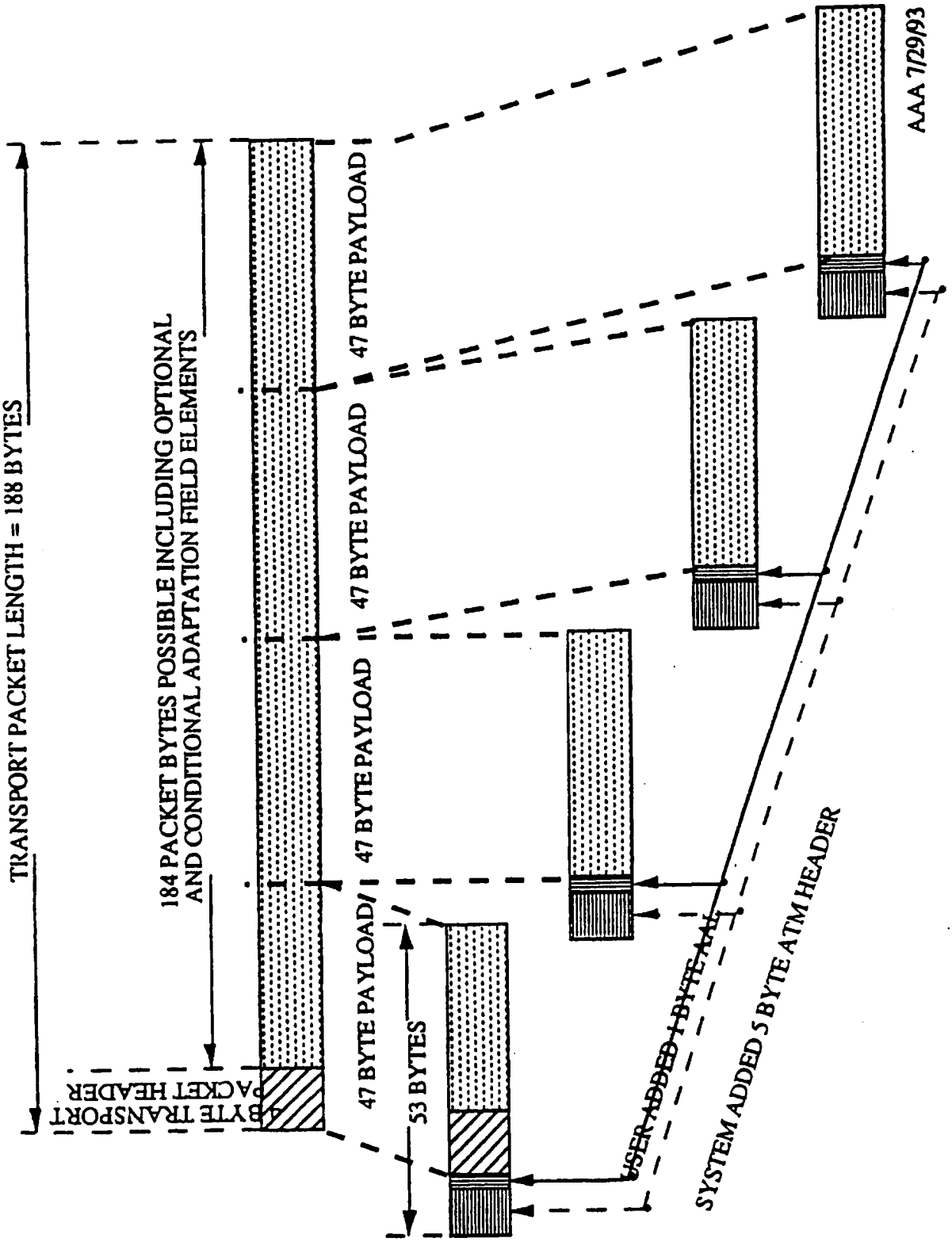


FIGURE 9.4. FORMATION OF THE ATM CELLS WITH A SINGLE AAL BYTE

INSDOCID: <XP_____2055175A_L>

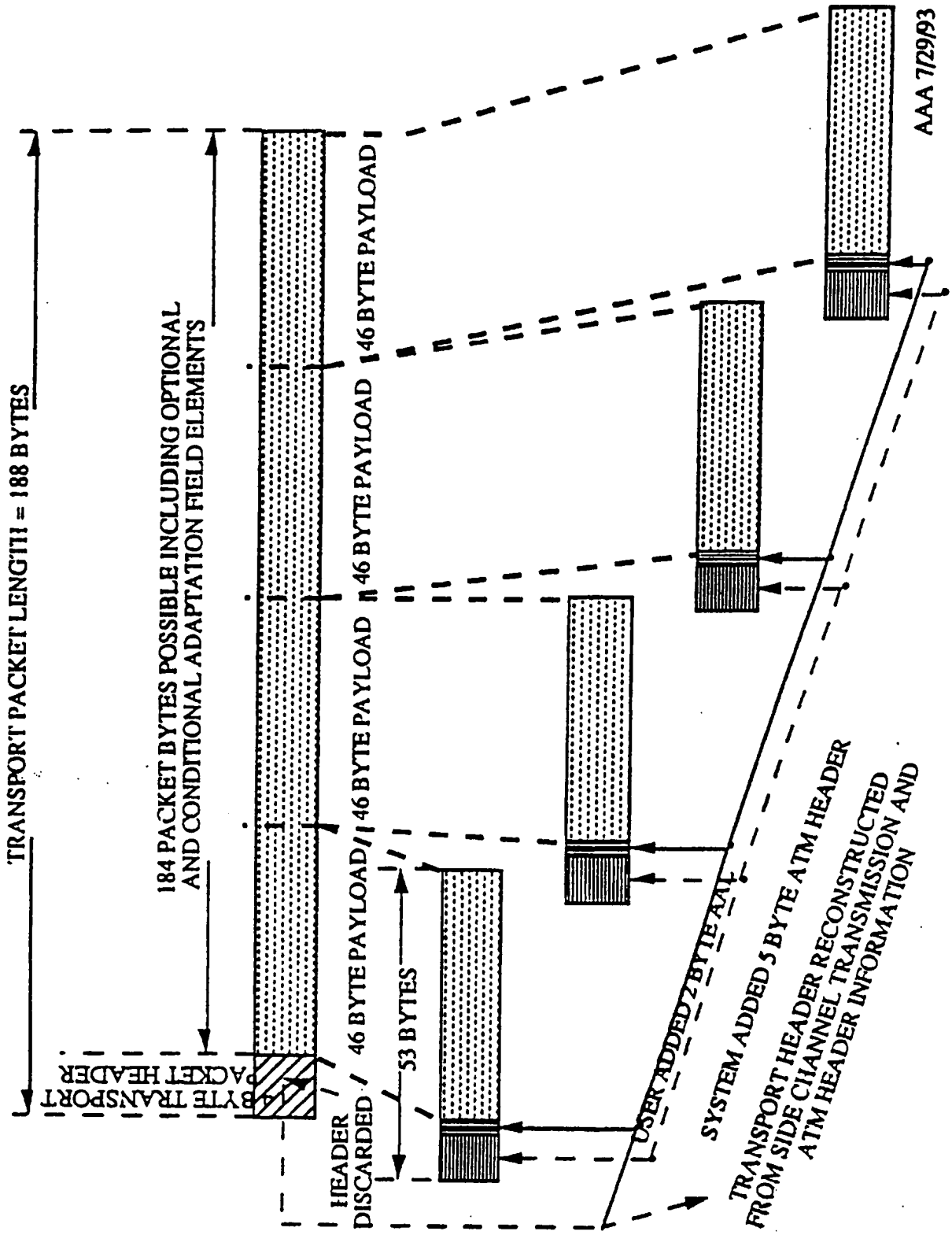


FIGURE 9.5. FORMATION OF AN ATM CELL WITH DUAL AAL

3NSDOCID: <XP_____2055175A_I_>